Jean-Daniel Boissonnat   Patrick Chenin
Albert Cohen   Christian Gout
Tom Lyche   Marie-Laurence Mazure
Larry Schumaker (Eds.)

LNCS 6920

Curves and Surfaces

7th International Conference, Curves and Surfaces 2010
Avignon, France, June 2010
Revised Selected Papers

Springer

Jean-Daniel Boissonnat   Patrick Chenin
Albert Cohen   Christian Gout   Tom Lyche
Marie-Laurence Mazure   Larry Schumaker (Eds.)

Curves and Surfaces

7th International Conference, Curves and Surfaces 2010
Avignon, France, June 24-30, 2010
Revised Selected Papers

Springer

Volume Editors

Jean-Daniel Boissonnat
INRIA Sophia-Antipolis, France
E-mail: jean-daniel.boissonnat@sophia.inria.fr

Patrick Chenin
Marie-Laurence Mazure
Université Joseph Fourier, Laboratoire LJK, Grenoble, France
E-mail:{marie-laurence.mazure, patrick.chenin}@imag.fr

Albert Cohen
Université Pierre et Marie Curie, Laboratoire Jacques-Louis Lions, Paris, France
E-mail: cohen@ann.jussieu.fr

Christian Gout
INSA Rouen, Laboratoire de Mathématiques, St. Etienne du Rouvray, France
E-mail: christian.gout@insa-rouen.fr

Tom Lyche
University of Oslo, CMA, Oslo, Norway
E-mail: tom@ifi.uio.no

Larry Schumaker
Vanderbilt University, Department of Mathematics, Nashville, TN, USA
E-mail: larry.schumaker@vanderbilt.edu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-27412-1 e-ISBN 978-3-642-27413-8
DOI 10.1007/978-3-642-27413-8
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011943775

CR Subject Classification (1998): I.3.5, I.4.8, G.1.1-2, F.2.2, G.2, J.6

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The 7th International Conference "Curves and Surfaces" was held June 24–30, 2010, at the Palais de Papes in Avignon, France. The conference was part of an ongoing joint French–Norwegian conference series on curves and surfaces. In France previous meetings were held in Chamonix in 1990, 1993, 1996, in Saint-Malo in 1999, 2002, and in Avignon in 2010. The last meeting in Norway was held in Tønsberg in 2008, and the proceedings have appeared as Volume 5862 of the *Lecture Notes in Computer Science*, published in 2009 by Springer.

The 2010 edition of Curves and Surfaces was attended by 261 participants from 35 different countries. The program included nine invited one-hour survey talks, eight minisymposia comprising 39 talks, 114 contributed talks, and two poster sessions with a total of 30 posters.

The conference was supported financially by Arts et Métiers Paris-Tech, the Center of Mathematics for Applications (CMA) at the University of Oslo, the Centre National de la Recherche Scientifique (CNRS), the Institut National de Recherche en Informatique et en Automatique (INRIA), the Université de Grenoble, and the Mairie d'Avigon.

We would particularly like to thank our irreplaceable webmaster Eric Nyiri for his work in supporting the Organizing Committee as well as the participants. His permanent good humor made these interactions easy and agreeable.

Our thanks are also due to our masterful TEX expert Yvon Lafranche who, as with all the earlier editions of our conference, has done a marvellous job of preparing the programme and the abstract booklet.

Would it have been possible even to imagine the conference without the skilful presence of Chantal Lyche? Her impressive performance at the head of the reception desk of the Palais des Papes was one of the key factors contributing to the smooth functioning of this event.

We are also indebted to all the others who helped with the conference, and in particular to Olivier Gibaru and Paul Sablonnière. A special mention goes to Michel Volle who, assisted by the University of Avignon, managed to solve certain delicate logistic issues.

Thanks are also due to all of the invited speakers and to all of the minisymposium organizers whose contributions were critical to making this conference a success. We would also like to thank all other presenters and participants.

Finally, we would like to express our gratitude to all of the reviewers who helped select articles for the present volume.

June 2010

Jean-Daniel Boissant
Patrick Chenin
Albert Cohen
Christian Gout
Tom Lyche
Marie-Laurence Mazure
Larry Schumaker

Table of Contents

Exact Medial Axis Computation for Triangulated Solids with Respect to Piecewise Linear Metrics

Oswin Aichholzer[1], Wolfgang Aigner[1], Franz Aurenhammer[2], and Bert Jüttler[3]

[1] Institute for Software Technology, Graz University of Technology, Austria
[2] Institute for Theoretical Computer Science, Graz University of Technology, Austria
[3] Institute of Applied Geometry, Johannes Kepler University Linz, Austria

Abstract. We propose a novel approach for the medial axis approximation of triangulated solids by using a polyhedral unit ball B instead of the standard Euclidean unit ball. By this means we compute the exact medial axis $\mathrm{MA}(\Omega)$ of a triangulated solid Ω with respect to a piecewise linear (quasi-) metric d_B. The obtained representation of Ω by the medial axis transform $\mathrm{MAT}(\Omega)$ allows for a convenient computation of the trimmed offset of Ω with respect to d_B. All calculations are performed within the field of rational numbers, resulting in a robust and efficient implementation of our approach. Adapting the properties of B provides an easy way to control the level of details captured by the medial axis, making use of the implicit pruning at flat boundary features.

Keywords: medial axis, piecewise linear metric, mesh boundary, trimmed offset.

1 Introduction

The medial axis is a skeleton-like structure, capturing the features of a shape in a lower-dimensional configuration. It has originally been introduced by Blum [7] for matters of shape representation, and has proved to be useful for various applications such as shape recognition, robot motion, finite element mesh generation [17], and offset computation. The computation of the exact medial axis – or of an approximation thereof – is a popular task in computational geometry and geometric computing. The huge variety of publications addressing different boundary representations [13,14,20], pruning techniques [9,22] and applications [8,12] is remarkable. See also [5] for a state of the art survey in this area. In the case of polyhedral objects, there exist numerical tracing techniques [24] (which have recently been extended to objects with curved boundaries [23]) and methods based on spatial decompositions [16,21].

For boundaries represented by dense point sets, it is a common approach to derive a medial axis approximation by isolating a subset of its Voronoi diagram [14]. The algorithm relies on heavy pruning and has (depending on the denseness of the point set) problems with capturing sharp features. Another approximating structure, that also allows to deal with non-exact boundaries, is the scale axis [22], based on a ball-representation of the shape. Pruning is achieved by careful scaling of the balls, which, on the downside, can lead to the introduction of topologically incorrect fragments. Both of these methods work in 2D and 3D, but they do not constitute an exact representation of a shape. They are thus suited for shape recognition and comparison but not for offset computation.

Exact medial axis computation relies on an exact boundary representation, and is well examined for piecewise smooth boundaries in 2-space. For straight-line polygons Lee [20] introduced an intuitive $O(n \log n)$ algorithm, which was later improved to an optimal (yet unimplemented) linear-time algorithm [11]. For circular arc boundaries a full implementation of a randomized algorithm (with expected $O(n \log n)$ computing time) is provided in [2]. To the contrary, it has turned out that in the three dimensional space the exact medial axis computation, even for shapes with piecewise linear boundaries, is a rather challenging problem. Here difficulties arise from the combinatorial complexity of the medial axis, as well as the high algebraic degree of its components. Especially the latter leads, due to the necessity of an algebraic kernel, to computing time and representation issues. So far, the only work in this context that provides a full implementation and some computing times is by Culver et al. [13], introducing complex algebraic algorithms to deal with the above-mentioned problems.

In this work we provide an approach that computes the exact medial axis of a triangulated solid (i.e., a solid object whose boundary surface is a triangular mesh) with respect to a piecewise linear quasi-metric d_B [26] induced by a convex polyhedral unit ball B (see also Minkowski functionals [19]). While the use of more general convex distance functions for bisector and Voronoi computation is no novelty [10,18], these generalized distances, however, have not been used for medial axis computations so far. This is quite surprising, considering that for given rational data (rational coordinates of mesh and unit ball vertices) the resulting linearity of the structure allows all computations to be performed within the field of rational numbers. We took advantage of this, providing a robust and stable implementation of the algorithm.

The quasi-metric d_B induces a piecewise linear medial axis transform $\mathrm{MAT}(\Omega)$, which describes the shape Ω fully and exactly, see Fig. 1a for an example. In order to deal with the structural complexity of the medial axis in 3D, we introduce planar contact arrangements, one for each possible contact between the components of the unit ball B and the boundary, respectively (see Sections 3 and 4). After computing these arrangements, we are able to calculate the components of the medial axis with respect to the quasi-metric d_B. In this way we reduce the problem of medial axis construction in 3D to a number of two dimensional problems.

The use of polyhedral unit balls permits interesting operations such as implicit pruning, resulting in pseudo-seams which will be introduced in Section 2. This allows us to influence the structure and complexity of the medial axis by varying combinatorial and geometrical properties of the unit ball. Furthermore, we will show that our representation via $\mathrm{MAT}(\Omega)$ is very convenient to compute trimmed offsets with respect to d_B, see Fig. 1b (see Section 5.2 for details).

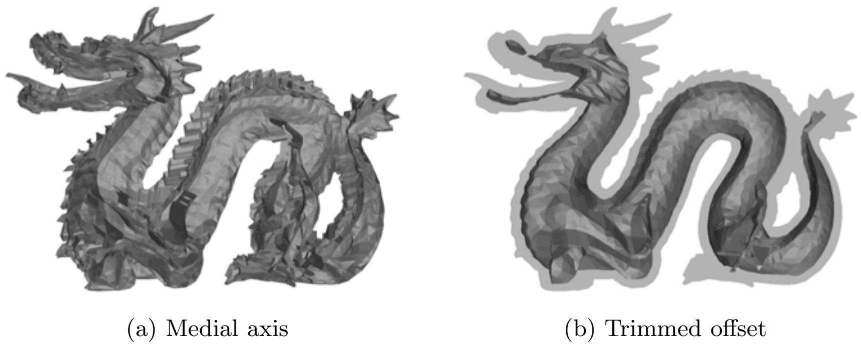


Fig. 1. Piecewise linear medial axis approximation and trimmed offset for a dragon mesh with 12,000 faces, using a quasi-metric defined by a tetrahedral unit ball

In Section 6 we will describe the close relation between the medial axes $\mathrm{MA}(\Omega)$ induced by Euclidean and polyhedral unit balls. This also identifies $\mathrm{MA}(\Omega)$ with respect to a piecewise linear metric d_B as an approximation of the Euclidean medial axis, where the quality of the approximation depends on the chosen unit ball B.

2 Preliminaries

Throughout this paper we consider an open set Ω in $\mathbb{R}^d$ ($d = 2, 3$) with a piecewise linear boundary $\partial\Omega$. We moreover assume that the boundary is triangulated and consists of edges, vertices, and triangular facets (the latter ones only for $d = 3$). We shall refer to Ω as a *triangulated solid.*

2.1 Unit Balls and Metrics

Let B be a bounded, open and convex set in $\mathbb{R}^d$ which contains the origin o. In particular, we are interested in two cases.

(E) B may be the usual *Euclidean unit ball*, $B = \{x : \ ||x|| < 1\} \subset \mathbb{R}^d$.
(L) B may be the interior of a *convex polyhedron*, i.e., the boundary ∂B is piecewise linear. Similar to $\partial\Omega$ we assume that ∂B is given by a triangulation.

In the second case (L) we shall assume that no edge or facet of ∂B is parallel to any edge or facet of $\partial\Omega$, i.e., we assume that Ω is in *general position* with respect to B. Later we will specify additional conditions that we assume to be satisfied.

By these assumptions it is guaranteed that a component of ∂B and a component of $\partial\Omega$ intersect in at most one point. To achieve this, a slight perturbation of the boundary of B and/or Ω – e.g. by application of the Simulation-of-Simplicity (SOS) technique [15] – can be applied. Clearly, by restricting the perturbation to the vertices of B we can even keep the original domain unchanged. However, even if perturbations are applied to the vertices of a triangulated solid, the resulting changes in the medial axis are not dramatic, provided that convex edges are not made reflex or vice versa.

For any points x and y, let r be the ray from x through y and B^* the body B translated by $\overrightarrow{ox}$. There exists a unique intersection point v of ∂B^* and r. The distance function

$$d_B(x,y) := \frac{\|y-x\|}{\|v-x\|} \tag{1}$$

defines a *quasi-metric* [26], meaning that d_B is positive definite and fulfills the triangle inequality, but is not necessarily symmetric. The given convex body B is the *unit ball* with respect to the quasi-metric.

If B is centrally symmetric with respect to the origin o, then d_B is a *metric*. In particular, the first choice of B as the Euclidean unit ball gives the usual Euclidean metric.

2.2 Maximal and Almost Maximal Balls

In the remainder of this paper we will use the symbols B', B'' etc. to represent convex polyhedra which are obtained from B by applying restricted Euclidean similarity transformations consisting of a *scaling* combined with a *translation*, but no rotation. Clearly, these convex sets are *balls* with respect to the quasi-metric defined by B, since they consist of all points whose distance d_B from the translated origin does not exceed the scaling factor.

Definition 1. A ball B' is said to be a *maximal ball* associated with the triangulated solid Ω if

1. it is contained in Ω, $B' \subseteq \Omega$, and if
2. any other ball B'' satisfying $B' \subset B''$ is not contained in Ω, i.e., $B'' \not\subseteq \Omega$.

Moreover, the ball B' is called an *almost maximal ball* associated with Ω, if it is contained in Ω and the boundary $\partial B'$ shares at least two points with $\partial \Omega$.

In the Euclidean case (E), the two notions are equivalent. In the case (L) of a piecewise linear metric, however, there may exist almost maximal balls which are not maximal.

2.3 Types of Contact

In this section we consider exclusively the case (L) of a piecewise linear metric.

If we consider a two-dimensional domain Ω in the plane, the following types of contact between $\partial\Omega$ and the boundaries $\partial B'$ of almost maximal balls are possible:

1. A vertex of $\partial\Omega$ is in contact with an edge of $\partial B'$, and
2. an edge or vertex of $\partial\Omega$ is in contact with a vertex of $\partial B'$.

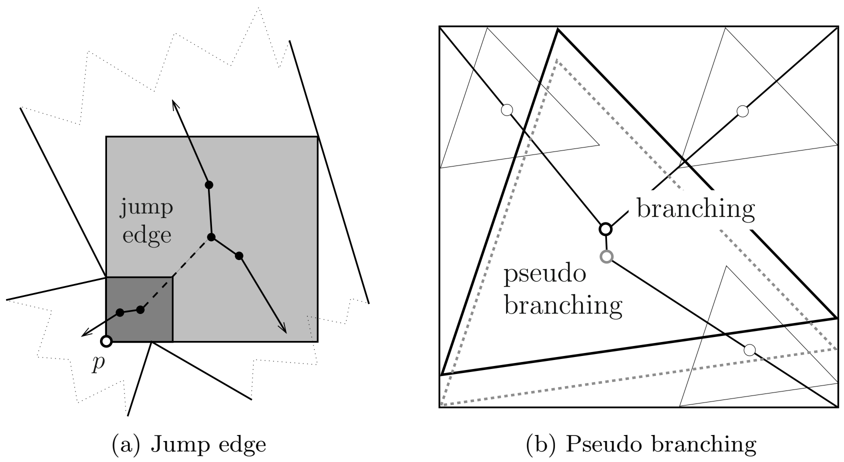


Fig. 2. (*a*) 2D example of a jump edge with center of scaling p. (*b*) A triangular unit ball induces a pseudo-branching in the medial axis of a square domain.

We will exclude the case where an almost maximal ball possesses two contacts of the first type that are realized at *only one edge* of $\partial B'$, by requiring that no edge of ∂B is parallel to any line connecting any two vertices of $\partial \Omega$. (It suffices to assume that this condition is satisfied by all pairs of non-convex vertices of $\partial \Omega$.) This is subsumed by the fact that we assume B and Ω to be in *general position*. We shall see later that almost maximal balls of this type would correspond to two-dimensional components of the medial axis.

Consider an almost maximal ball B' that possesses exactly two contacts which are of the first type and realized in the interior of two *neighboring* edges of $\partial B'$, and let p be the common vertex of the neighboring edges. In this case, any uniform scaling with a factor f sufficiently close to 1 and center p transforms B' into another almost maximal ball which is either a subset (if $f < 1$) or a super-set (if $f > 1$) of B', see Fig. 2a.

The same phenomenon occurs if an almost maximal ball B' possesses contacts of the first and the second type, and the contact vertex of $\partial B'$ is a segment end point of the contact edge of $\partial B'$.

In the three–dimensional case, the following types of contact between $\partial \Omega$ and the boundaries $\partial B'$ of almost maximal balls are possible:

1. A vertex of $\partial \Omega$ is in contact with a facet of $\partial B'$,
2. a vertex or an edge of $\partial \Omega$ is in contact with an edge of $\partial B'$, and
3. a vertex, an edge or a facet of $\partial \Omega$ is in contact with a vertex of $\partial B'$.

Again we exclude the case of almost maximal balls with two contacts of the first type which are realized in the interior of *only one facet* of $\partial B'$, and the case of almost maximal balls with two contacts of the second type at two coplanar edges of $\partial \Omega$ which are realized in the interior of *only one edge* of $\partial B'$, by assuming that Ω and B are in *general position*.

Similar to the discussion in the planar situation one may observe that an almost maximal ball B' with only two contacts that are realized at two *neighboring* entities (i.e., facets, edges, or vertices) of $\partial B'$ is not maximal, since it is possible to apply a uniform scaling with a center that is located in the intersection of the two contact entities.

2.4 Medial Axis

We define the *medial axis* $\mathrm{MA}(\Omega)$ as the union of the centers of all almost maximal balls associated with Ω. The *medial axis transform* $\mathrm{MAT}(\Omega)$ additionally contains the information about the scaling of the almost maximal balls which are centered at the points of $\mathrm{MA}(\Omega)$.

The medial axis of a *planar shape* Ω consists of bisector curves (edges) and trisector points (branching points). In the general (non-degenerate) case, three edges meet at a branching point.

Consider the case (L) of a piecewise linear metric. Here, some of the bisectors correspond to nested families of almost maximal balls, which share the same contacts of type 1 on the boundary. These bisectors will be called *jump edges*, since the maximal inscribed balls jump between the two extreme positions, see Fig. 2a. If we did not consider jump edges, using only truly maximal balls for the definition, the medial axis of a connected planar domain Ω would possibly consist of several disconnect components. Moreover, if we relaxed the assumption of the general position by allowing almost maximal balls with two contacts of type 1 in the interior of only one edge of $\partial B'$, these balls would produce *two-dimensional* components of the medial axis.

The medial axis of a *three-dimensional domain* Ω consists of bisector surfaces (sheets), trisector curves (seams) and junctions. In the generic case – meaning that there do not exist maximal balls with more than four contacts on the boundary of Ω – three sheets meet at a seam, and four seams meet at a junction point [13]. For the case (L), similar to the case of jump edges for planar domains, some of the sheets correspond to partially nested families of almost maximal balls. We will refer to them as *jump sheets*. Once again, these jump sheets – and consequently the consideration of almost maximal balls – are needed in order to guarantee that the medial axis of connected domains is again connected. By relaxing the assumption of general position one would obtain three-dimensional components of the medial axis, which do not occur in the Euclidean case and thus are clearly not desirable.

Proposition 1. *The medial axis in the case (L) is a piecewise linear structure.*

Proof. The bisectors of linear structures with respect to a piecewise linear metric or quasi-metric are again linear structures. The medial axis of a triangulated solid with respect to such a metric is composed of these bisectors and their intersections, which are also linear. □

Another new phenomenon that occurs when using a piecewise linear metric (L) instead of the Euclidean one (E) is the implicit pruning of convex features (edges or vertices) of the boundary, which are flat with respect to the unit ball, in the sense that a vertex of the unit ball fits into the wedge defined by the feature. Such features lead to the appearance of special branching points (see Fig. 2b) or seams, which we will call *pseudo-branchings* and *-seams*, respectively. The almost maximal balls centered there share only two points with the boundary of Ω. In the planar case, one of these contacts has to be of type vertex-vertex. In the 3D case, one of these contacts is of the type edge-vertex or vertex-edge.

We will come back to this issue in the next section.

3 Contacts and Contact Arrangements

In the next three sections we consider solely the case of piecewise linear metric (L) in three-dimensional space. All arguments are easily adaptable to the planar case.

3.1 Contacts

Recall that a ball B' is a scaled and translated copy of the polyhedral unit ball B. We shall denote the vertices, edges and facets of ∂B, $\partial B'$, and Ω uniformly as *components* of these boundaries.

For any boundary component x of B, we denote with x' its image under the restricted similarity transformation (translation and scaling) that maps B to B'. Moreover, for each boundary component x of ∂B we choose an arbitrary but fixed *representative* vertex $v = v(x)$, which is one of the three vertices of a triangle, one of the two end points of an edge, or the vertex itself in the case of a vertex.

Since we assumed that B and Ω are in general position, every boundary component (vertex, edge or facet) of an almost maximal ball shares at most one point with a component of $\partial\Omega$.

Definition 2. Consider an almost maximal ball B' and assume that the component y of $\partial\Omega$ has a common point with the component x' of B'. We say that the pair (x, y) is a *contact.*

The *regular* combinations of boundary components – which determine the structure of the medial axis – are vertex-facet contacts, edge-edge contacts and facet-vertex contacts. Even for objects and unit balls in general position, vertex-edge, edge-vertex and vertex-vertex contacts do occur, but they can be regarded as being *singular*. They define pseudo-structures of the medial axis, but do not induce any sheets or seams.

An almost maximal ball with *two* contacts is centered on a *sheet* of the medial axis, a ball with *three* contacts on a seam (cf. Fig. 3). An almost maximal ball centered on a *pseudo-seam* is also defined by *three* contacts, where two of these contacts are *adjacent*, meaning that the ball components, as well as the mesh components, are incident, respectively. As a consequence, the almost maximal

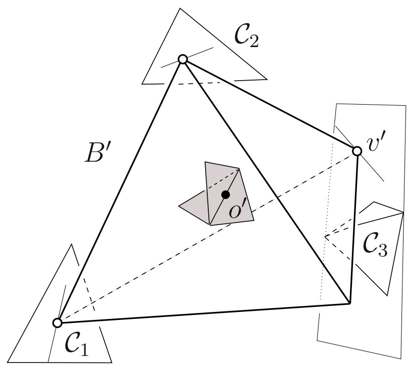


Fig. 3. The center o' of an almost maximal ball B' lies on a seam of the axis. The point v' is its projection on the contact plane of $\mathcal{C}_3$.

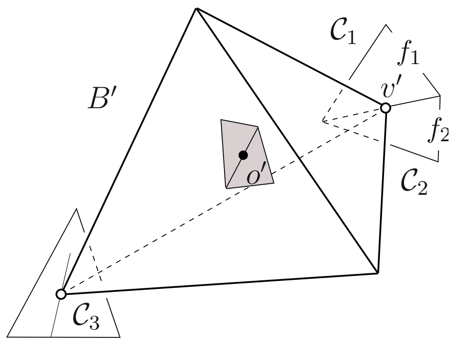


Fig. 4. The contact $\mathcal{C}_1 = (v, f_1)$ is adjacent to $\mathcal{C}_2 = (v, f_2)$. Consequently, the three contacts define a pseudo-seam containing o'.

ball's contact that is induced by these two adjacent contacts is, dependent on their types, of type vertex-edge or edge-vertex (see Fig. 4 for an illustration).

For every possible contact (x, y) the component $y \in \partial\Omega$ and the transformed ball component $x' \in B'$ span a plane, which will be called the *contact plane* associated with the contact.

3.2 Projections

An almost maximal ball B' possesses at least two contacts. Let $v(x)$ be the representative vertex of the ball part x of one contact (x, y) among them. We call v', i.e., the equivalent of v on the translated and scaled copy B' of the unit ball B, the *projection* of the center o' into the contact plane of (x, y).

Definition 3. Given a contact (x, y), let $\mathcal{B}'(x, y)$ be the set of all almost maximal balls which realize this contact (x, y). The set of all projections of the balls in $\mathcal{B}'(x, y)$ into the contact plane describes a polygonal region on the contact plane of (x, y). We will call $\mathcal{D}(x, y)$ the *contact domain* of (x, y).

A contact domain is the union of projections of medial axis components on the contact plane. As these components are piecewise linear, so are the projections on the plane and their union. Therefore a contact domain is a polygonal region.

Roughly speaking, the contact domain $\mathcal{D}(x, y)$ describes the trace of the representative vertex v for all almost maximal balls B' which share the contact (x, y). For a vertex-facet contact (x, y), the contact domain is contained in the mesh facet y, and there is only one contact with this facet. For the other non-singular types of contacts, the domain is contained in a plane containing the boundary component, and there may be several contacts sharing a boundary component. A more detailed discussion will be given in [3]. The singular contacts (vertex-vertex, edge-vertex, and vertex-edge contacts) do not define a two-dimensional domain.

3.3 Contact Arrangements

A seam of the medial consists of the center points o' of almost maximal balls B' that possess the same three contacts. For each of these three contacts (x, y), the projections of the centers o' into the contact plane define a line segment on the contact plane, see Fig. 3. This line segment is contained in the contact domain $\mathcal{D}(x, y)$. In a similar way we obtain line segments that are projections of pseudo-seams.

The projections of all seams and pseudo-seams that share a given contact (x, y) form an arrangement of line segments, which we will call the *contact arrangement*, in the contact domain $\mathcal{D}(x, y)$.

Every edge of the contact arrangement represents a seam or a pseudo-seam. The junction points of the medial axis correspond to the vertices of the contact arrangement.

Remark 1. The medial axis may possess *jump sheets*, which correspond to partially nested families of almost maximal balls. While general sheets of the medial axis correspond to two-dimensional parts of the contact arrangements, the jump sheets may be represented by one-dimensional components (i.e., edges) as well, by choosing the representative vertex in a suitable way. Therefore, we need to treat jump sheets in a special way. This will be described in more detail in [3].

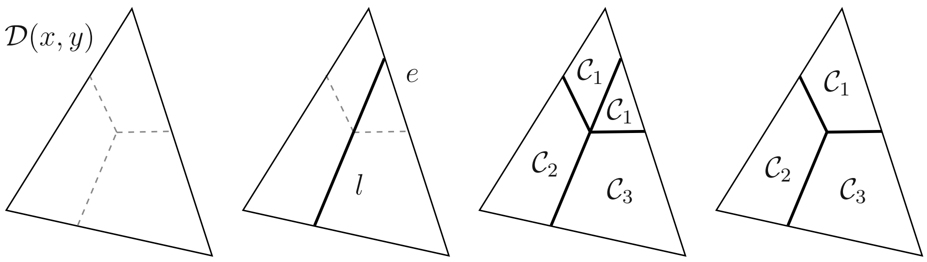
4 Computing the Contact Arrangements

As an almost maximal ball is implicitly defined by its contacts, the medial axis is fully represented by the contact arrangements. In order to analyze the medial axis, we compute the contact arrangements for all possible contacts (x, y). Consequently, we reduce the problem of medial axis computation to a finite number of two-dimensional problems in the respective contact planes, which can moreover be addressed in parallel, since they are mutually independent.

4.1 Outline of the Algorithm

For each contact (x, y) and its contact plane P, we perform the following algorithm, which is summarized visually in Fig. 5.

1. Create a stack of subdomains lying in P, and initialize it with the entire contact domain.
2. If the stack is empty, then continue with step 4, otherwise take a subdomain from the stack.
3. Check if there exists a seam or pseudo-seam which defines a projection line segment in P that hits the subdomain. If such a projection line is found, then split the subdomain along the line spanned by the segment into new subdomains and add them to the stack. Continue with the previous step.
4. Remove all line segments in the arrangement that do not represent projections of seams or pseudo-seams.



(a) Contact domain (b) Projection line (c) Clean subdomains (d) Arrangement

Fig. 5. Computation of a contact arrangement in a contact plane P

4.2 Constructing Almost Maximal Balls

Once again, let $v = v(x)$ be the representative vertex of the contact (x, y). The most frequent (and also most expensive) operation of the algorithm is to compute an almost maximal ball B' for a point p on the contact domain, such that v' and p coincide. In particular, it is crucial to identify the remaining contacts of such an almost maximal ball. If there is only one additional contact, then p lies on a face of the contact arrangement, otherwise, it belongs to an edge.

An almost maximal ball is found by iterative shrinking, where p is the center of scaling. We start with a ball satisfying $p = v'$ which is sufficiently large to intersect the boundary mesh (see Fig. 6a). With help of an AABB (Axis Aligned Bounding Box) tree [4], the intersections between components of the ball boundary and the mesh are efficiently detected. The component of the mesh closest to p determines the shrinking factor. This is done iteratively until the shrunk ball and the mesh are intersection-free (see Fig. 6b). The last component of the mesh which is used to define the shrinking induces the second contact of the almost maximal ball. As all the above computations are done within the set of rational numbers, the resulting almost maximal ball and its center point are exact.

4.3 Finding Projection Lines

A projection line in the contact domain – which may be determined by a seam or a pseudo-seam – always corresponds to a change of the second contact of the associated almost maximal balls. Thus, the projection lines subdivide the contact domain into subdomains whose points define almost maximal balls with the same second contact.

Consider two points p and q on a contact domain. If the two associated almost maximal balls have different second contacts then we know that there exists a projection line crossing $\overline{pq}$. On the other hand, if the balls share the same opposite contact, then this does *not* imply that there is no such crossing line, since the faces of the contact arrangement are not necessarily convex.

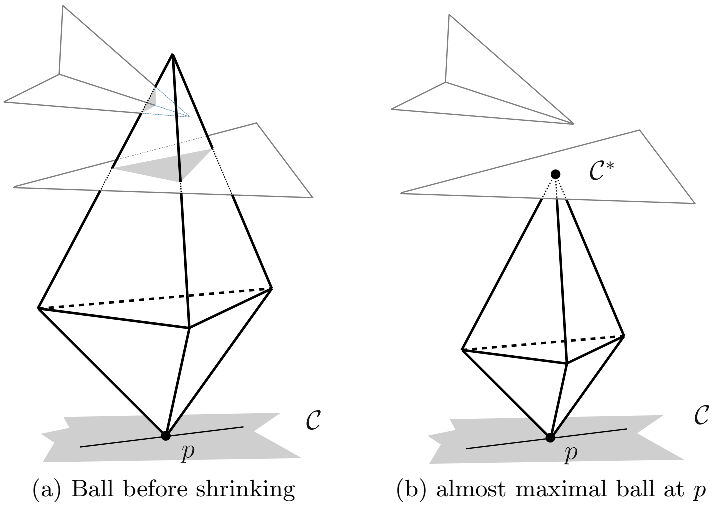


Fig. 6. Iterative computation of an almost maximal ball with a given projection p in a given contact plane (x, y). The second contact is $\mathcal{C}^*$.

If the associated second contacts $\mathcal{C}_p$ and $\mathcal{C}_q$ of p and q are different we need to find a point on the segment $\overline{pq}$ which lies on a projection line. Roughly speaking, this is achieved by constructing a ball based on $\overline{pq}$ and confined by the contact planes of $\mathcal{C}_p$ and $\mathcal{C}_q$. If this ball turns out to be a valid almost maximal ball of Ω with three contacts, then its center lies on a seam or pseudo-seam and induces a projection line. Otherwise the interval between q and p is split and the search for two opposite contacts that define a projection line is continued iteratively by binary search. In non-singular configurations, this process is guaranteed to terminate.

On the other hand, in order to verify that no projection line crosses the edge $\overline{pq}$, where p and q have the same second contact, the family of almost maximal balls along $\overline{pq}$ (which spans a convex polyhedron) has to be contained in Ω, see again [3] for more details. If the line segment $\overline{pq}$ is not crossed by any projection line, then we call this segment *clean*. The final subdomains of the contact arrangement are characterized by the fact that they are bounded by clean segments.

4.4 Summary

For any given point on the contact domain we can construct the associated almost maximal ball. For two points on the domain we can decide if there exists a projection line that crosses the connecting segment of the points, and eventually find such a line. This is all we need to build the contact arrangement.

We start with the complete contact domain, and iterate over its boundary segments (Fig. 5a). If one of the segments induces a projection line, we split the domain at this line into two new subdomains and continue recursively (e.g., the edge e induces projection line l in Fig. 5b). If all boundary edges of a subdomain are *clean*, then the subdomain is *clean* and all points contained in it are associated with almost maximal balls having the same opposite contact, and thus lie on the same sheet of the axis (Fig. 5c). When all subdomains are clean we remove all artifact edges between neighboring subdomains describing the same sheet (two faces with opposite contact $\mathcal{C}_1$ are merged in Fig. 5d). This finally gives us the contact arrangement.

Remark 2. As said in Remark 1, a jump sheet may, depending on the representative vertex, correspond to a one-dimensional projection on a contact plane. Such a special *jump projection edge* is detected by an algorithm similar to the one for seams and pseudo-seams, which is, however, a bit more involved. For the computation of the contact arrangement such an edge is handled like any other projection line. For other representative vertices the jump sheet corresponds to a two-dimensional component (i.e., face) of the arrangement. In this case no *jump projection edge* occurs. For more details in this context see [3].

5 Assembling the Medial Axis and Offset Computation

Once we have computed all contact arrangements, the medial axis can be assembled by a simple algorithm. Based on this result we address the problem of trimmed offset computation. Finally we report experimental results that indicate the relation between the complexity of the input data (number of facets on $\partial\Omega$ and ∂B), the computing times and the size of the generated output.

5.1 Assembling the Medial Axis from Its Projections

When all contact arrangements are computed, the assembling of the axis can be performed by a simple computation. Any sheet of the axis is associated with two faces of two different contact arrangements, a seam with three edges of three arrangements. A pseudo-seam is induced by one arrangement edge, and two segments on the domain boundaries of two neighbored contacts. A jump sheet is associated with a *jump projection edge* or a face of a contact arrangement, depending on the representative vertex chosen for this contact. Every vertex of the arrangement is associated with an almost maximal ball B', and the center points o' span the medial axis.

The resulting medial axis is a non-manifold connected piecewise linear mesh. Connectivity can in general be derived from the contact arrangements. This means that two axis components are incident if their projections are incident in a contact arrangement. The *radial edge structure* introduced in [25] is one of several data structures that recommends itself for storing such a non-manifold mesh.

As a first example we consider a slightly perturbed octahedron Ω and compute its medial axis with respect to several polyhedral unit balls B, where the number of facets increases from 4 to 128. The results are shown in Fig. 7.

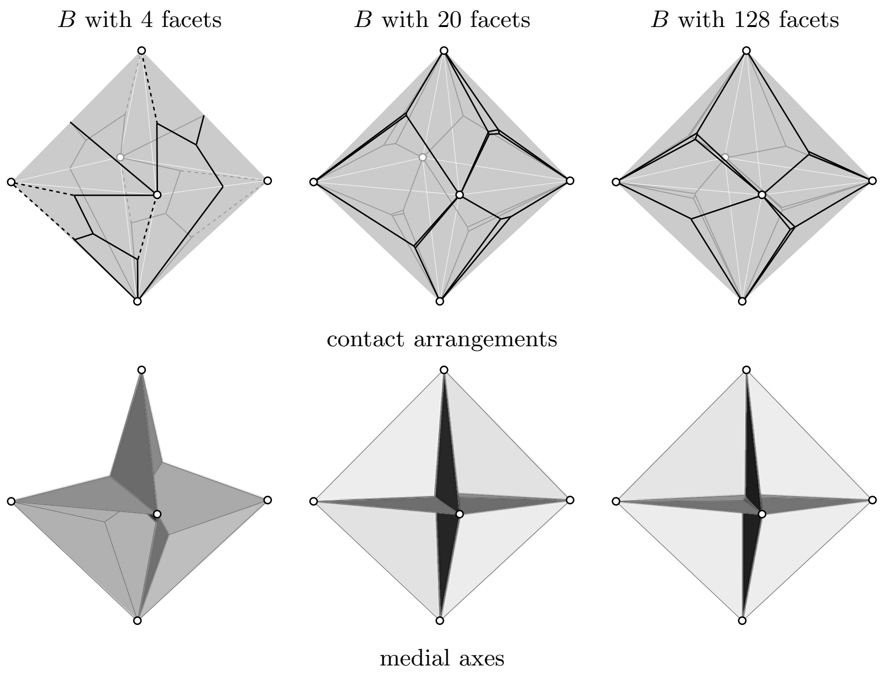
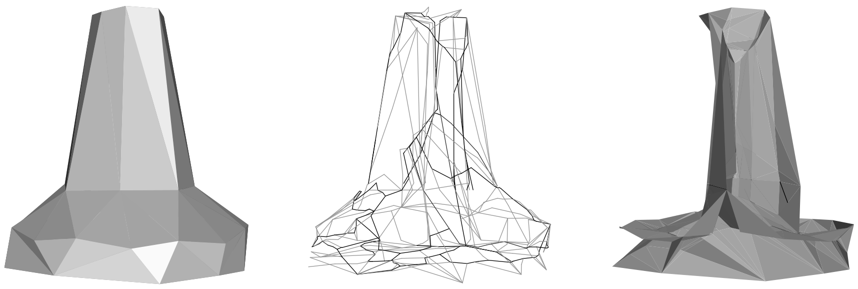


Fig. 7. Contact arrangements (top row) and medial axes (bottom row) of a slightly perturbed octahedron with respect to polyhedral unit balls with 4, 20 and 128 facets (from left to right). Dashed lines are projections of pseudo-seams.

Since Ω is convex in this example, all contact domains are contained in the facets of Ω and only vertex-face contacts need to be considered. Consequently, the projections and contact arrangements can be visualized directly on $\partial\Omega$ (shown in the first row). The medial axis of the octahedron Ω with respect to the Euclidean unit ball consists of three squares which intersect each other along their diagonals. The medial axis with respect to a sparse polyhedral unit ball (a tetrahedron) is quite different (bottom left), since some of the vertices of the ball fit into the edge and vertex wedges of of the domain. When using a a polyhedral unit ball with a larger number of facets (bottom center and right), however, the structure of the computed medial axis is quite similar to the Euclidean case.

As a second example, we consider the "tower" object. Fig. 8 shows the object, the contact arrangements (projections) and the medial axis with respect to a piecewise linear quasi-metric generated by a tetrahedron. The mesh consists of 80 triangular facets and the resulting medial axis counts 269 sheets. As the object is non-convex, not all projections are realized directly on its boundary.



(a) Tower mesh (b) Projections (c) Medial axis

Fig. 8. Tower mesh, projections, and medial axis. The grey lines in (b) are the projections of pseudo-seams.

5.2 Offset Computation

The medial axis is a useful tool for trimmed offset computation. While this is well-established in the two-dimensional case [1,8], the structure has not yet been used much in 3-space for this purpose [6].

The medial axis representation which is generated by our algorithm is directly useful for offset computation with respect to a linear (quasi-)metric. Each sheet $\mathcal{S}$ of the medial axis is associated with two contacts $\mathcal{C}_1$ and $\mathcal{C}_2$. An almost maximal ball B' with center point o' on $\mathcal{S}$ and scaling factor s' has a unique point of contact p_i on $\mathcal{C}_i$ for $i \in \{1, 2\}$. Let ρ be the offset size. Then the offset operation with ρ applied to B' gives us a new point p_i^ρ for each of the two contacts. This new point p_i^ρ lies on the line defined by p_i and o'. The position of p_i^ρ with respect to the sheet $\mathcal{S}$ determines whether or not it has to be trimmed:

- If $s' > \rho$ then p_i^ρ lies between p_i and o'. Therefore p_i^ρ is a valid point of the offset surfaces.
- If $s' < \rho$ then o' lies between p_i and p_i^ρ. Therefore p_i^ρ has to be trimmed.
- If $s' = \rho$ then $p_1^\rho = p_2^\rho = o'$ and the point lies on the axis sheet where the trimmed and valid part of the offset surfaces are joined.

The axis sheets as well as the assigned faces of the contact arrangements are polyhedral regions. A triangulation on the sheet induces a triangulation on the faces, leaving us with a configuration as visualized in Fig. 9, where the three almost maximal balls at the corner points are known. Depending on the offset size ρ, certain parts of the triangles that lie on planes parallel to the contact planes define the valid offset surface. Note that a part derived from an edge-edge or facet-vertex contact resides on a plane which is partially defined by features of the unit ball.

We define the trimmed offset in 3D analogously to the planar one in [1]. It should be noted that the obtained offset is induced by the distance function

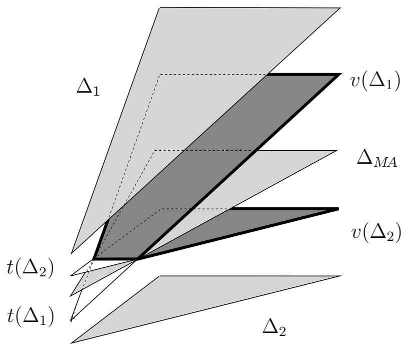


Fig. 9. The triangle Δ_{MA} of an axis sheet induces two triangles Δ_1 and Δ_2 on two different contact arrangements. The offset surface generated from each of these triangles is split into a valid ($v(\Delta)$) and a trimmed ($t(\Delta)$) part, which intersect in the corresponding axis sheet.

d_{-B}, where $-B$ is the image of B under reflection at the origin o. Clearly, the two distance functions d_B and d_{-B} are identical for centrally symmetric unit balls B.

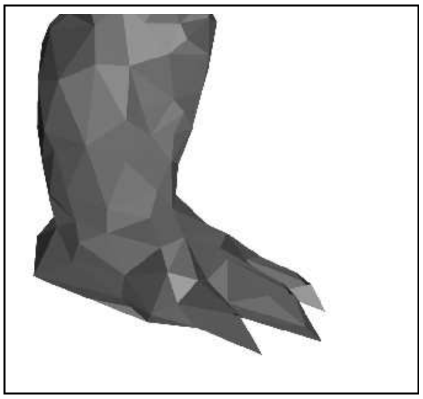
We performed the trimmed offset computation for the Armadillo mesh. A typical result is shown in Fig. 10.

5.3 Computing Time and Size of the Medial Axis

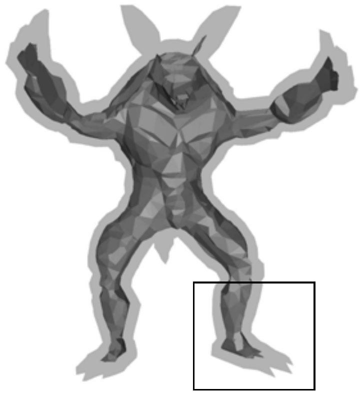
The time needed for the computation of the contact arrangements depends on various criteria. The quality of the boundary mesh influences the computing time gain provided by the AABB-tree structure. A rather complex and strongly branched shape has more reflex features and thus more edge-edge and facet-vertex contacts. On the other hand the nesting complexity of the single contact arrangements is in average higher for less ramified shapes, which also increases the computing time.

At this stage we cannot present any theoretical results. In order to obtain empirical data, we used several instances of the Armadillo mesh (see Fig. 10), and tested it against various polyhedral unit balls (see Fig. 11). The computation times are reported in Fig. 12. They, as well as the ones provided in Table 1 for several instances of the “Venus”-shape, compare favorably with the ones reported in [13], which is the only implementation we are aware of that constructs the exact medial axis with respect to a specific metric. There, the computation of the medial axis for the “Venus”-shape with 250 faces is performed in 5.6 hours, with computing times growing considerably with respect to the number of faces. As can be seen in Table 1, we compute the exact medial axis with respect to the quasi-metric induced by a tetrahedral B for an instance with 267 faces (see Figure 14) in less than 5 minutes. Also, the computation times for the Armadillo and the Venus example grow only slightly super-linearly with respect to the

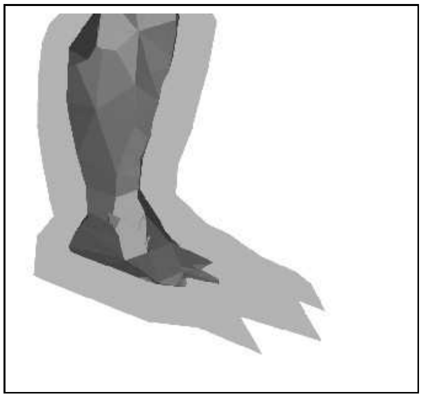
(a) Armadillo mesh with $3,124$ facets



(b) Mesh detail



(c) Offset for tetrahedral B

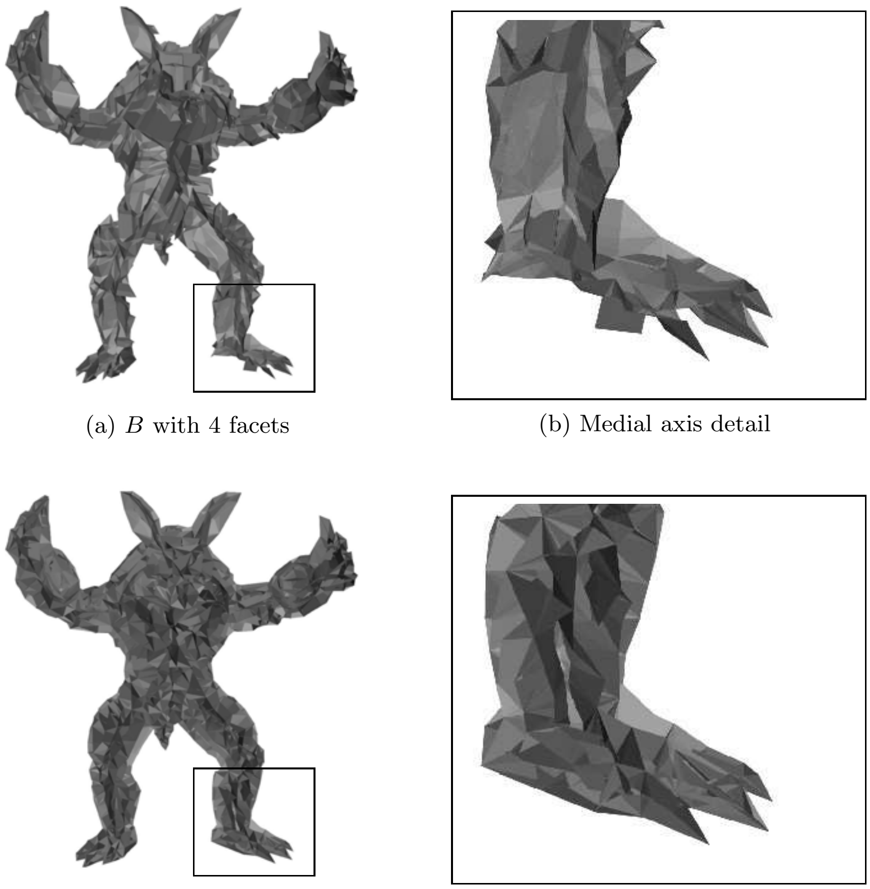


(d) Offset detail

Fig. 10. A version of the Armadillo mesh with 3124 facets and its trimmed offset for d_B with respect to a tetrahedral unit ball B

number of facets of the mesh, and even sub-linearly with respect to the number of facets in the unit ball.

Finally we analyze the relation between the size (i.e., the number of planar sheets) of the computed medial axis and the number of facets on the boundaries of $\partial\Omega$ and of ∂B, see Fig. 13, again for the Armadillo example. The size of the medial axis grows linearly with the size of $\partial\Omega$, but only very slowly (much less than linear) with the size of ∂B. This will be analyzed in more detail in the future.



(a) B with 4 facets

(b) Medial axis detail

(c) B with 128 faces

(d) Medial axis detail

Fig. 11. The medial axis of the armadillo mesh from Fig. 10 for two different unit balls B

6 Convergence

The quasi-metric defined by a convex polyhedron B can be seen as an approximation of the Euclidean metric. Indeed, if the unit ball B converges to the Euclidean unit ball, then the quasi-metric defined by it converges to the Euclidean metric. The convergence of the unit balls can be described with the help of the Hausdorff distance. Recall that the Hausdorff distance of two sets X and Y is defined as

$$\mathrm{HD}(X,Y) = \max(\sup_{x\in X}\inf_{y\in Y}\|x-y\|, \sup_{y\in Y}\inf_{x\in X}\|x-y\|). \tag{2}$$

# faces	**4**	**8**	**20**	**128**
96	0.03	0.05	0.15	1.28
194	0.05	0.12	0.34	2.63
390	0.10	0.24	0.66	5.05
780	0.21	0.51	1.43	10.13
1562	0.43	0.95	2.44	19.17
3124	0.85	1.74	4.67	34.24
6250	1.66	3.34	8.40	58.09
12500	3.37	6.23	15.39	101.12
25000	7.32	12.39	28.13	170.69
50000	18.28	27.46	57.38	-
100000	52.30	66.80	124.09	-

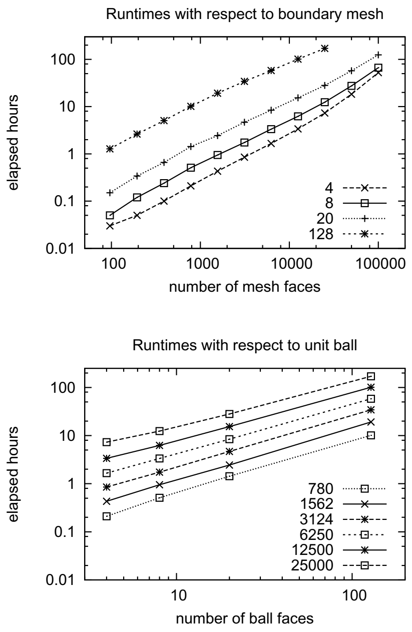


Fig. 12. Left: Computation times (in hours) for several polyhedral unit balls (shown in the different columns; the first row specifies the number of faces) and various instances of the Armadillo mesh (shown in the rows) on a single CPU with 2.5 GHz. Right: Results plotted on a log-log scale.

# faces	**4**	**8**	**20**	**128**
96	315	325	375	542
194	661	714	819	1097
390	1410	1437	1709	2315
780	2879	3154	3661	4945
1562	6106	6689	7316	10091
3124	12514	13365	15043	20655
6250	24764	26519	29841	39906
12500	48592	52655	58055	78155
25000	94715	101733	111355	148967

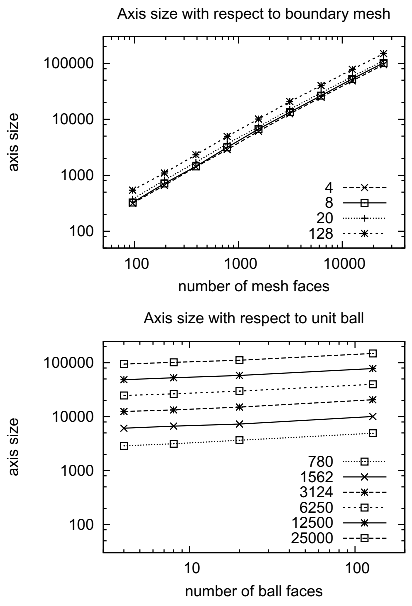


Fig. 13. Number of sheets of the medial axis for several polyhedral unit balls (shown in the different columns; the first row specifies the number of faces) and various instances of the Armadillo mesh (shown in the rows). Right: Results plotted on a log-log scale.

Table 1. Computation times in seconds for different combinatorial sizes of B (rows) and different instances (columns) of the Venus model shown in Figure 14

# faces	**4**	**8**	**20**	**128**
115	115.54	270.44	834.66	4645.03
267	284.04	594.75	1846.20	11770.50
575	525.19	1117.43	3011.68	21619.20
1396	1209.51	2120.49	6005.36	37837.30

In this section we consider simultaneously two metrics and the associated medial axes. On the one hand, we have the piecewise linear (quasi-) metric d_B defined by the convex polyhedron B and the medial axis $\mathrm{MA}_B(\Omega)$ of the given domain Ω with respect to it. On the other hand, we have the usual Euclidean metric and the standard medial axis, which we will now denote with $\mathrm{MA}(\Omega)$.

6.1 Planar Domains

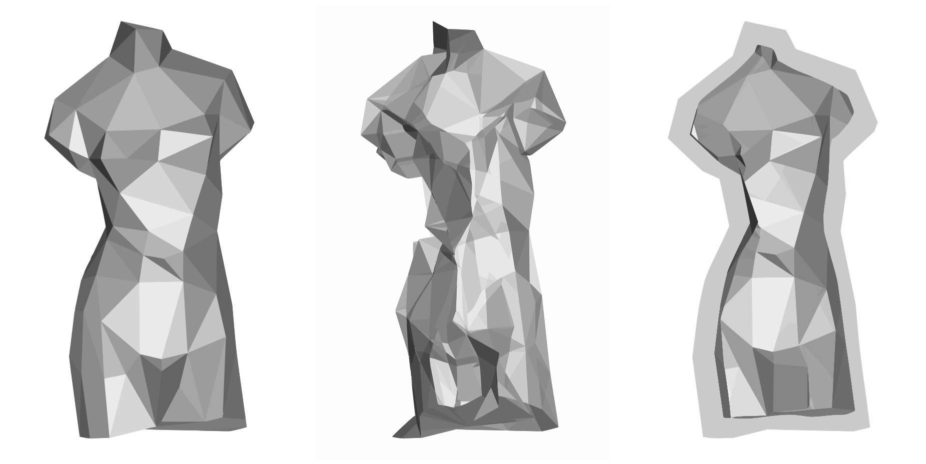
For planar domains $\Omega \subset \mathbb{R}^2$, the following result establishes a close connection between the two skeletal structures $\mathrm{MA}_B(\Omega)$ and $\mathrm{MA}(\Omega)$:

Theorem 1. *Consider a planar domain $\Omega \subset \mathbb{R}^2$ with piecewise linear boundary $\partial\Omega$. If the convex polygon B that serves as the unit ball of the (quasi-) metric d_B converges to the Euclidean unit circle, then the Hausdorff distance between the medial axes $\mathrm{MA}_B(\Omega)$ and $\mathrm{MA}(\Omega)$ with respect to the piecewise linear (quasi-) metric and the Euclidean metric, respectively, tends to zero.*

Thus, the convergence of the unit ball implies the convergence of the medial axis. Before proving this result we present the following result, which is visualized in Fig. 15.

Lemma 1. *Each point c' of the medial axis $\mathrm{MA}_B(\Omega)$ of the planar domain Ω sees any two of its associated closest points on the boundary under a certain angle $\alpha'(c')$. For polygonal unit balls B that are sufficiently close to the Euclidean unit circle, there exists a lower bound φ' of this angle, which is independent of B. Each point c of the medial axis $\mathrm{MA}(\Omega)$ sees any two of its associated closest points on the boundary under a certain angle $\alpha(c)$. There exists a lower bound φ of this angle.*

Proof. First we observe that none of the almost maximal polyhedral balls B' has a contact with the boundary of Ω in a convex vertex, provided that B is sufficiently close to the Euclidean unit circle. Similarly, none of the maximal Euclidean balls touches the boundary of Ω in a convex vertex. Consequently, each almost maximal polyhedral ball B' and each maximal Euclidean ball has contact



(a) Mesh (b) Medial axis (c) Trimmed offset

Fig. 14. (*a*) A mesh instance of the Venus model with 267 faces. (*b*) The medial axes induced by a unit ball B with 4 faces. (*c*) The resulting trimmed offset.

– with two edges of $\partial\Omega$ (both contacts are of type 1),
– with an edge and with a reflex (non-convex) vertex of $\partial\Omega$ (the contacts are of type 1 and type 2), or
– with two reflex vertices of $\partial\Omega$ (both contacts are of type 2).

Some balls may have more than two contacts, but we need to consider only two of them.

In the latter two cases, we consider the minimum distance d between any two reflex vertices and between any reflex vertex and any edge not starting or ending at this vertex. For all points c corresponding to these two types of contact, the angle $\alpha(c)$ satisfies $\alpha(c) \geq 2\arcsin(d/D)$, where D is the diameter of Ω (which is also an upper bound on the diameter of the maximal Euclidean circles). Consequently, if B is sufficiently close to the Euclidean unit circle, the angle $\alpha'(c')$ satisfies $\alpha'(c') \geq \arcsin(d/D)$.

In the first case, the two contacts are realized at two non-parallel edges of $\partial\Omega$. Let β be the smallest angle between any two non-parallel edges of $\partial\Omega$. Here we consider all pairs of edges, not just the adjacent ones. For all points c corresponding to this type of contact, the angle $\alpha(c)$ satisfies $\alpha(c) \geq \beta$. Consequently, if B is sufficiently close to the Euclidean unit circle, the angle $\alpha'(c')$ satisfies $\alpha'(c') \geq \beta/2$. $\square$

Now we are ready to prove the convergence result.

Proof (Theorem 1). First we consider a point $c' \in \mathrm{MA}_B(\Omega)$ and prove that there exists a point $c \in \mathrm{MA}(\Omega)$ such that $\|c' - c\| \leq \varepsilon(B)$, where $\varepsilon(B)$ tends to zero as B converges to the Euclidean unit ball.

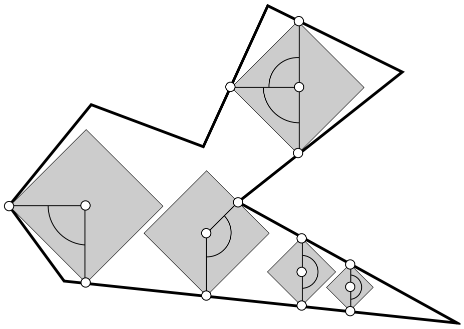
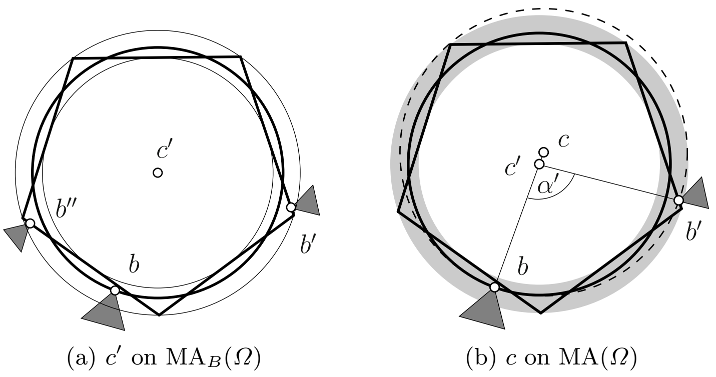


Fig. 15. Each center of a maximal (or almost maximal) ball sees any two of its associated boundary points under a certain angle. For piecewise linear boundaries, which are in general position with respect to the unit ball, this angle has a lower bound.



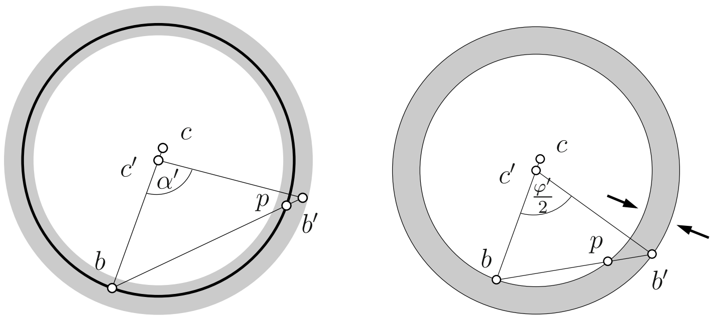
c'
b''
b
b'
(a) c' on $\mathrm{MA}_B(\Omega)$
c
c'
α'
b
b'
(b) c on $\mathrm{MA}(\Omega)$

Fig. 16. The almost maximal polyhedral ball with center c' (left) and the construction of the associated maximal Euclidean ball (dashed) with center c (right)

For a given $c' \in \mathrm{MA}_B(\Omega)$ we consider the associated almost maximal ball B', along with its inscribed circle and circumscribed circle. The almost maximal ball B' touches the boundary in at least two points $b', b'' \in \partial\Omega$, see Fig. 16a.

Consider the largest inscribed Euclidean ball with center c'. It touches the boundary $\partial\Omega$ at a point b, which is generally different from both b' and b''. The boundary of this Euclidean ball lies between the inscribed circle and the circumscribed circle. The center c' sees b and one of the other two points – say b' – under an angle $\alpha > \frac{\varphi'}{2}$.

We consider the maximal inscribed Euclidean ball which is obtained by applying uniform scaling with center b and scaling factor $1 + \delta$ to the ball with center c', see Fig. 16b. This scaling maps the center c' into a new center c satisfying



(a) b, p and b' (b) The bound δ_0

Fig. 17. The construction of an upper bound on the scaling factor $1 + \delta$

$$\|c - c'\| = \delta\|c' - b\| \leq \delta D \tag{3}$$

where D is the diameter of the domain Ω. We find an upper bound on δ by considering the intersection p of the line segment from b to b' with the Euclidean circle with center c'. The uniform scaling moves this point towards b', but not beyond b', hence

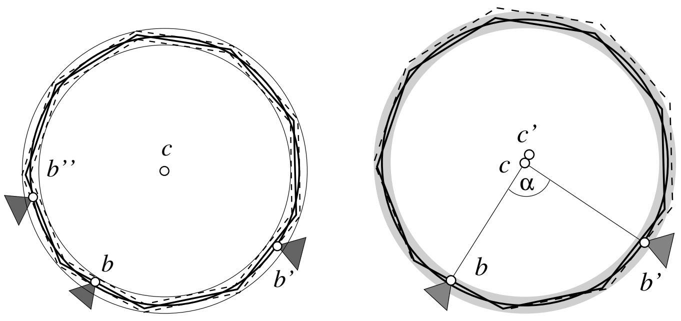
$$\delta < \frac{\|p - b'\|}{\|p - b\|}, \tag{4}$$

cf. Fig. 17a. This upper bound on δ remains valid if the following operations are applied: First, we apply a uniform scaling with center c' and a scaling factor ≤ 1 which moves b and p to the inscribed circle. Second, we shift b' to the circumscribed circle along the line bb'. Third, the enclosed angle $\angle bc'b'$ is reduced to $\frac{\varphi'}{2}$, see Fig. 17b.

Let δ_0 be the upper bound on δ obtained after these operations, i.e., from the configuration in Fig. 17b. We can bound the distance $\|c-c'\|$ by $\varepsilon(B) = \delta_0(B)D$. Finally, if the polyhedral ball B converges to the Euclidean ball, then the distance between the inscribed and the circumscribed circle shrinks. Consequently, we obtain $\delta_0(B) \to 0$ and hence $\varepsilon(B) \to 0$.

In the second part of the proof we consider a point $c \in \mathrm{MA}(\Omega)$ and prove that there exists a point $c' \in \mathrm{MA}_B(\Omega)$ such that $\|c' - c\| \leq \varepsilon'(B)$, where again $\varepsilon'(B)$ tends to zero as B converges to the Euclidean unit ball. This can be proved by swapping the roles of circles and polyhedral balls with respect to the Euclidean and the piecewise linear metric, as follows.

For a given $c \in \mathrm{MA}(\Omega)$ we consider the associated maximal Euclidean ball, along with its inscribed piecewise linear circle and circumscribed piecewise linear circle. This situation is visualized in Fig. 18a. The inscribed and circumscribed piecewise linear circles are shown as dashed polygons.



(a) c on $\mathrm{MA}(\Omega)$ (b) c' on $\mathrm{MA}_B(\Omega)$

Fig. 18. The maximal Euclidean ball with center c (left) and the construction of the associated almost maximal polyhedral ball (dashed) with center c' (right)

The inscribed piecewise linear circle possesses an inscribed Euclidean circle, and the circumscribed piecewise linear circle possesses a circumscribed Euclidean circle. We will now refer to these two Euclidean circles as the inscribed circle and the circumscribed circle, respectively.

The maximal Euclidean ball with center c touches the boundary in at least two points $b', b'' \in \partial\Omega$, see Fig. 18a. Consider the largest inscribed piecewise linear ball with the same center c. It touches the boundary $\partial\Omega$ at a point b, which is generally different from both b' and b''. The boundary of this Euclidean ball lies between the inscribed circle and the circumscribed circle. The center c sees b and one of the other two points – say b' – under an angle $\alpha > \frac{\varphi}{2}$.

Similar to the first part of the proof we consider the almost maximal inscribed piecewise linear ball which is obtained by applying uniform scaling with center b and scaling factor $1+\delta'$ to the piecewise linear ball with center c, see Fig. 18b. This scaling maps the center c into a new center c' satisfying

$$\|c - c'\| = \delta'\|c - b\| \le \delta' D. \tag{5}$$

As in the first part of the proof we are now able to construct an upper bound δ'_0 on δ'. If the polyhedral ball B converges to the Euclidean ball, then the distance between the inscribed and the circumscribed circle shrinks, which again implies $\delta'_0(B) \to 0$, and hence $\varepsilon'(B) \to 0$.

Finally, by combining the results of both parts we see that the Hausdorff distance of $\mathrm{MA}(\Omega)$ and $\mathrm{MA}_B(\Omega)$ tends to zero as B converges to the Euclidean unit ball. □

Let h denote the Hausdorff distance between B and the Euclidean ball. The upper bounds δ_0 and δ'_0 can be bounded by Ch, where the constant C depends on the angles φ and φ'. Consequently, the Hausdorff distance of $\mathrm{MA}(\Omega)$ and $\mathrm{MA}_B(\Omega)$ is bounded by CDh. The constant C, however, is rather large for small values of φ and φ'.

6.2 Towards a Convergence Proof for the 3D Case

In order to extend this approach to the spatial case, it is first necessary to analyze the possibility of generalizing Lemma 1. Unfortunately, for triangulated solids in 3D, it turns out that no such lower bound the angles α and α' exists in general.

If the piecewise linear ball is sufficiently close to the Euclidean one, then it suffices again to consider only piecewise linear balls that do not fit into any of the convex edges of the domain. Each almost maximal piecewise linear ball and each maximal Euclidean ball of Ω has contact

– with two facets of $\partial\Omega$, or
– with two entities of $\partial\Omega$, where at least one of them is a reflex edge or a non-convex vertex.

In the first case, a lower bound on the angles φ and φ' can be derived as in the planar case. In the second case, this is possible only if the two contact entities do not possess any common points.

More precisely, if the two entities which are present in the second case are two reflex edges with a common vertex, or a reflex edge and a facet possessing a common vertex, then the technique used for proving the result in the planar case can no longer be applied, since it requires a lower bound on the distance between the two entities. Thus, a more sophisticated approach is required in order to generalize the convergence result to the 3D case.

We expect that the following approach allows to extend Theorem 1 to triangulated solids in space. First, we consider only the subset of the medial axes which are generated by almost maximal balls and by maximal Euclidean balls where the angle introduced in Lemma 1 exceeds a certain threshold ϕ^*. We denote these subsets by $\mathrm{MA}^*_B(\Omega)$ and $\mathrm{MA}^*(\Omega)$, respectively.

Next we consider a sequence $(B_n)_{n=1,2,\ldots}$ of unit balls with the property that the ratio between the radii of the circumscribed and the inscribed ball has the upper bound $1 + 1/n^3$. For each of these balls we use the associated threshold $\phi^*_n = 1/n$ to define the subsets $\mathrm{MA}^*_B(\Omega)$ and $\mathrm{MA}^*(\Omega)$. Thus, after an appropriately scaling of B_n, the Hausdorff distance between the piecewise linear unit ball and the unit ball tends to zero as $1/n^3$, while the lower bound on the angle tends to zero as $1/n$.

Using the same techniques as in the proof of Theorem 1, we can conclude that the one-sided Hausdorff distances between $\mathrm{MA}^*_B(\Omega)$ and $\mathrm{MA}(\Omega)$ and between $\mathrm{MA}^*(\Omega)$ and $\mathrm{MA}_B(\Omega)$ converge to zero as $n \to \infty$. Simultaneously, the lower bound ϕ^*_n on the angle ϕ used for defining $\mathrm{MA}^*_B(\Omega)$ and $\mathrm{MA}^*(\Omega)$ tends to zero.

Finally, it should be possible to prove that the Hausdorff distances between $\mathrm{MA}^*(\Omega)$ and $\mathrm{MA}(\Omega)$, and between $\mathrm{MA}^*_B(\Omega)$ and $\mathrm{MA}_B(\Omega)$ converge to zero as well. The desired convergence result can then be obtained by combining these observations. The details of this proof cannot be described satisfactorily in the frame of this paper and will be reported elsewhere.

7 Concluding Remarks

We have presented an algorithm which computes a piecewise linear medial axis representation $\mathrm{MA}(\Omega)$ of a triangulated polyhedron Ω with respect to a piecewise linear quasi-metric d_B. The representation allows convenient trimmed offset computation, and all computations can be performed within the field of rational numbers. We would like to point out that the shape is not required to be simply connected, as the axis-representing contact arrangements are computed independently. This makes the algorithm easily accessible for parallel implementation. The algorithm shows convincing computational complexity and is suitable for larger meshes.

The complexity of the polyhedral unit ball can be chosen depending on the respective application. This is also one of the interesting issues for future research in this area. Given a mesh, what does a (preferably combinatorially small) polyhedral unit ball have to look like to reduce the occurrence of pseudo-seams? With a decreasing number of pseudo-seams, a combinatorial structure close to the Euclidean medial axis is to be expected. On the other hand the implicit pruning induced by the piecewise linear metric might be a welcome feature. This leads to the question how to locate points on the unit sphere, such that the vertices of the resulting convex polyhedral ball enter as many flat convex features of a mesh as possible.

Modifications of the unit ball B do affect the geometric as well as the combinatorial appearance of $\mathrm{MA}(\Omega)$. Another interesting task is to identify and isolate the combinatorially stable – and thus essential – parts of the medial axis by comparing the representations for different quasi-metrics d_B resulting from several different polyhedral unit balls B.

Acknowledgment. This research was supported by the Austrian Science Fund (FWF) through the National Research Network S92 "Industrial Geometry", subprojects 2 and 5. The authors thank the reviewers for their comments which have helped to improve the paper.

References

1. Aichholzer, O., Aigner, W., Aurenhammer, F., Hackl, T., Jüttler, B., Pilgerstorfer, E., Rabl, M.: Divide-and-conquer for Voronoi diagrams revisited. Comput. Geom. Theory Appl. 43, 688–699 (2010)
2. Aichholzer, O., Aigner, W., Aurenhammer, F., Hackl, T., Jüttler, B., Rabl, M.: Medial axis computation for planar free-form shapes. Comput. Aided Des. 41, 339–349 (2009)
3. Aigner, W.: Generalized representation of geometric objects in 2D and 3D. PhD thesis, Graz University of Technology (2011) (in preparation)

4. Alliez, P., Tayeb, S., Wormser, C.: AABB Tree. In: CGAL User and Reference Manual. CGAL Editorial Board (2010)
5. Attali, D., Boissonnat, J.-D., Edelsbrunner, H.: Stability and computation of medial axes: a state of the art report. In: Hamann, B., Moeller, T., Russell, B. (eds.) Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration. Mathematics and Visualization, Springer, Heidelberg (2007)
6. Bastl, B., Jüttler, B., Kosinka, J., Lávička, M.: Volumes with piecewise quadratic medial surface transforms: Computation of boundaries and trimmed offsets. Computer-Aided Design 42, 671–679 (2010)
7. Blum, H.: A transformation for extracting new descriptors of shape. In: Wathen-Dunn, W. (ed.) Models for the Perception of Speech and Visual Form, pp. 362–380. MIT Press, Cambridge (1967)
8. Cao, L., Liu, J.: Computation of medial axis and offset curves of curved boundaries in planar domain. Comput. Aided Des. 40(4), 465–475 (2008)
9. Chazal, F., Lieutier, A.: The λ-medial axis. Graph. Models 67, 304–331 (2005)
10. Chew, L.P., Kedem, K., Sharir, M., Tagansky, B., Welzl, E.: Voronoi diagrams of lines in 3-space under polyhedral convex distance functions. J. Algorithms 29, 238–255 (1998)
11. Chin, F., Snoeyink, J., Wang, C.A.: Finding the medial axis of a simple polygon in linear time. In: Discrete Comput. Geom., pp. 382–391. Springer, Heidelberg (1995)
12. Chung, J.-M., Ohnishi, N.: Matching and recognition of planar shapes using medial axis properties (2007)
13. Culver, T., Keyser, J., Manocha, D.: Exact computation of the medial axis of a polyhedron. Comput. Aided Geom. Des. 21, 65–98 (2004)
14. Dey, T.K., Zhao, W.: Approximating the medial axis from the Voronoi diagram with a convergence guarantee. Algorithmica 38, 179–200 (2003)
15. Edelsbrunner, H., Mücke, E.P.: Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. ACM Trans. Graph. 9(1), 66–104 (1990)
16. Etzion, M., Rappoport, A.: Computing the Voronoi diagram of a 3-D polyhedron by separate computation of its symbolic and geometric parts. In: Proceedings Symposium on Solid Modeling and Applications, pp. 167–178. ACM, New York (1999)
17. Gursoy, H.N., Patrikalakis, N.M.: An automatic coarse and finite surface mesh generation scheme based on medial axis transform. part 1: algorithms. Engineering with Computers 8, 121–137 (1992)
18. Icking, C., Klein, R., Lê, N.-M., Ma, L., Santos, F.: On bisectors for convex distance functions in 3-space. In: Proc. 11th Canad. Conf. Comput. Geom. (1999)
19. Kelley, J.L., Namioka, I.: Linear Topological Spaces. Springer, New York (1976)
20. Lee, D.T.: Medial axis transformation of a planar shape. IEEE Trans. Pattern Analysis and Machine Intelligence 4, 363–369 (1982)
21. Lee, Y.-G., Lee, K.: Computing the medial surface of a 3-D boundary representation model. Advances in Engineering Software 28, 593–605 (1997)
22. Miklos, B., Giesen, J., Pauly, M.: Discrete scale axis representations for 3D geometry. ACM Trans. Graph. 29, 101:1–101:10 (2010)

23. Ramanathan, M., Gurumoorthy, B.: Interior medial axis transform computation of 3D objects bound by free-form surfaces. Computer-Aided Design 42, 1217–1231 (2010)
24. Sherbrooke, E.C., Patrikalakis, N.M., Brisson, E.: An algorithm for medial axis transform of 3-D polyhedral solids. IEEE Trans. on Visualization and Computer Graphics 2, 44–61 (1996)
25. Weiler, K.: The radial-edge structure: A topological representation for non-manifold geometric boundary representations. In: Geometric Modelling for CAD Applications, pp. 3–36 (1988)
26. Wilson, W.A.: On quasi-metric spaces. American Journal of Mathematics 53(3), 675–684 (1931)

Exact Medial Axis Computation for Circular Arc Boundaries

Oswin Aichholzer[1], Wolfgang Aigner[1], Thomas Hackl[1], and Nicola Wolpert[2]

[1] Institute for Software Technology, Graz University of Technology, Austria
[2] Faculty Geomatics, Computer Science and Mathematics,
Stuttgart University of Applied Sciences, Germany

Abstract. We propose a method to compute the algebraically correct medial axis for simply connected planar domains which are given by boundary representations composed of rational circular arcs. The algorithmic approach is based on the Divide-&-Conquer paradigm, as used in [2]. However, we show how to avoid inaccuracies in the medial axis computations arising from a non-algebraic biarc construction of the boundary. To this end we introduce the Exact Circular Arc Boundary representation (ECAB), which allows algebraically exact calculation of bisector curves. Fractions of these bisector curves are then used to construct the exact medial axis. We finally show that all necessary computations can be performed over the field of rational numbers with a small number of adjoint square-roots.

Keywords: medial axis, circular boundary, exact computation.

1 Introduction

The *medial axis* is an important concept for shape description introduced by Blum [4]. We call a domain S in the plane a simple shape, if it is bounded by a non-selfintersecting closed curve ∂S. The medial axis of S is composed of the union of all center points of *maximal disks* inscribed in S. If S is simple then its axis has a tree-like structure. The following two definitions stem from [4]:

Definition 1. *Given a shape S, a disk $D \subseteq S$ is called maximal, if there does not exist a disk $D' \subseteq S$, $D' \neq D$, which contains D. We denote the set of all maximal disks with $MAT(S)$ (medial axis transform of S).*

Definition 2. *Given a shape S, its medial axis $MA(S)$ is defined as the union of all centers of maximal disks in S:*

$$MA(S) := \{c_D \mid D \in MAT(S), \text{ and } c_D \text{ is the center of } D\} \ .$$

The medial axis construction for shapes with simple boundary representations as straight lines or circular arcs is a field that has been tackled with various techniques. The Divide-&-Conquer approach used in [2] is a simple method for efficient axis computation, however, with some minor drawbacks. The biarc construction as described in [2] provides theoretical smoothness, that is however not representable by usual float or rational number types. Furtheron, degenerate branching points of the medial axis cannot be detected exactly. In particular, the correct representation of the medial axis curve is a challenging task if the boundary input data does not comply with certain (numerical or algebraic) quality criteria as being rational representable or providing algebraically smooth joints between arcs.

An important part of most medial axis algorithms is the bisector computation. This problem has been approached for various types of rational curves, but mostly relying on machine arithmetic as in [11,7].

Our goal is to compute the algebraically correct medial axis. Thus, we have to cope with exact bisector computation of (arc-supporting) circles. For this purpose we require all arcs on the boundary, that are involved in the bisector computation, to be *rational.* Arcs which do not directly contribute to the medial axis, but describe a local curvature maximum and thus merely a leaf-point of the axis, are allowed to be rational square-root expressions (*rasqex*).

Integers are rasqex. If x and y are rasqex, so are $x+y$, $x-y$, $x \cdot y$, x/y and $\sqrt{x}$. Rasqex have exact comparison operators $=$, $<$, and $>$, realized in LEDA [6,14] or the Core library [13,16]. Actually, these two packages are able to represent arbitrary k-th root numbers, what is more than we need. For our purposes the *FieldWithSqrt* concept as provided by the CGAL library [1] is sufficient.

Several details of our algorithm, e.g. bisectors and tritangent circles, are similar to those needed for the construction of an Apollonius diagram, as examined extensively in the work of Emiris and Karavelas [8]. They show that the operations allowed in the rasqex number type are sufficient to compute all predicates. Similar efforts have been made for ellipses and even more general smooth convex sites [9,10]. Beside the similarities there are serveral additional aspects we have to take into account for our approach. First, the medial axis construction needs parts from the underlying bisectors different from the ones needed for the Apollonius diagram. Second, while in [8] an incremental approach is pursued, we intend to show that all steps of our Divide-&-Conquer algorithm can be accomplished with rasqex numbers as well, a fact that is not obvious. Furthermore, as opposed to the Apollonius diagram, we do not deal with single sites and complete circles, but one closed curve composed of circular arcs representing a planar shape. In this context we consider boundaries that are at least C^1-smooth to define an *Exact Circular Arc Boundary* or *ECAB.* (The extension to non-smooth boundaries only requires an extension of the cases that may occur for bisector computation.) See Section 2 for detailed definitions.

Given an *ECAB*, the divide-part of the algorithm presented in [2] is applied (overview in Section 3) with some minor modification of the construction of the *dividing disks* (Section 4), as they are crucial for the final reassembling of the medial axis. The main bisector calculus takes place when arriving at the base cases which terminate the decomposition process. Pairs of (rational) arcs are adequately chosen, and the bisectors of their supporting circles are computed.

It is shown in Section 5 that the bisectors are algebraic curves of degree 4 over the rational numbers $\mathbb{Q}$, which can be expressed as the product of two quadratic polynomials (conics) over a simple extension field of $\mathbb{Q}$. The center points of the arcs stemming from the dividing disks (called *artificial arcs*) lie on the bisectors, and are used to isolate those parts of the conic curves which contribute to the partial medial axes of specific base cases. Details about these base cases are discussed in Section 6, representing the conquer-part of the algorithm.

2 Exact Circular Arc Boundary

We define a circular boundary representation which fits our needs for an exact bisector and medial axis construction. This requires some definitions, starting with *rational circles*. Let $\mathbb{Q}$ denote the set of rational numbers.

Definition 3. *For a circle C with center c the following definitions are equivalent:*

$$\begin{aligned} C \text{ is a rational circle} &\iff c \in \mathbb{Q}^2 \quad \text{and} \quad \exists u \in C : u \in \mathbb{Q}^2 && (1)\\ &\iff \exists u, v, w \in \mathbb{Q}^2 : u, v, w \in C\ . && (2)\end{aligned}$$

Note that the squared radius of a rational circle is rational. It is also well-known that on a rational circle C points with rational coordinates are lying dense (see [15]). This means that near an arbitrary point p on C and for any $\epsilon > 0$ one can find a rational point in an ϵ-environment around p, that lies on C. We say that an arc is *rational*, if its supporting circle and its two endpoints are rational. By extending to rasqex numbers, we can now define *rasqex circles* as a superset of rational circles.

Definition 4. *For a circle C with center c and squared radius r the following definitions are equivalent:*

$$C \text{ is a rasqex circle} \iff c \text{ and } r \text{ are rasqex}\ .$$

An arc is *rasqex*, if its supporting circle and its two endpoints are rasqex. A rational circle is always a rasqex one, but not vice versa. For our C^1-boundary representation we want to rely on rational circles as much as possible, but to build a C^1-smooth boundary consisting exclusively of rational arcs means a severe restriction. We therefore soften our demands by allowing rasqex arcs whenever they are not directly contributing to bisector calculation. This is true for arcs which describe a *local curvature maximum*, as such a maximum always defines a leaf-point of the medial axis which just represents the endpoint of a medial axis curve. This means such an arc does not contribute to any bisector computation later on (Section 5), only its center point is eventually required for point location (Section 6). According boundary construction rules are given in Section 7.

Definition 5. *Consider a C^1-circular arc boundary representation. An arc that constitutes a local curvature maximum, and thus a leaf point of the medial axis, has to be at least rasqex. If all other arcs are rational, then we call this structure an Exact Circular Arc Boundary (ECAB).*

Note that the restriction to C^1 is not a necessary one. For general circular arc boundaries however we have to deal with more base cases and with bisector construction between points and circles. This does not pose any problems concerning computation, but is left aside for reasons of lucidity.

3 The Divide-and-Conquer Algorithm

We are interested in computing the medial axis for a given ECAB boundary. As our approach is closely related to the work introduced in [2] we first give an overview here. The algorithm is based on the Divide-&-Conquer paradigm. The dividing step consists of finding a random disk $D \in MAT(S)$ (called *dividing disk*) and decomposing S, with the help of D, into sub-shapes. The latter is done by splitting the boundary, adding *artificial arcs* that originate from D, and rearranging the *original arcs*. After this has been applied recursively, down to predefined base cases, the merge step is a simple concatenation of the partial axes at the center points of the dividing disks, as they are guaranteed to lie on the original medial axis. For a list of base cases that may occur for C^1-smooth boundaries, see Figure 3. We leave the basic algorithmic approach almost unchanged. But with the help of the properties of the ECAB structure (as opposed to numerical biarc constructions), and by modifying a few specific steps, a mathematically correct representation of the medial axis is made possible:

- The construction of a dividing disk has to be done with care, to take advantage of the properties of rational arcs. The centers of the dividing disks play a more important role, as they are required to lie exactly on the bisector curves for segment confinement. This is discussed in Section 4.
- The bisector computation is now done in an algebraic way, avoiding any numerical errors. The whole computation can be done over the field of rational numbers with only a few adjoint square-roots, as shown in Section 5.
- The handling of the base cases is more sophisticated, however the algebraic approach allows us to detect degenerate constellations more easily (Section 6).

4 Constructing Dividing Disks

Dividing disks, being maximal disks as defined in Section 1, are required for the recursive decomposition of a shape S. A general maximal disk has two contact points on ∂S, which lie on two different arcs when dealing with an ECAB.

As in [2] we start by choosing a random arc p on the boundary. The only limitation is that p must not define a local curvature maximum, meaning it does not induce a leaf-point of the medial axis. The ECAB structure then tells us that p is rational and thus we can choose a rational point t_p as close to an arbitrary point on p as we see fit. See [5] for a detailed algorithm and implementation on how to choose such a point. For every arc $q \neq p$ of ∂S we construct the disk that

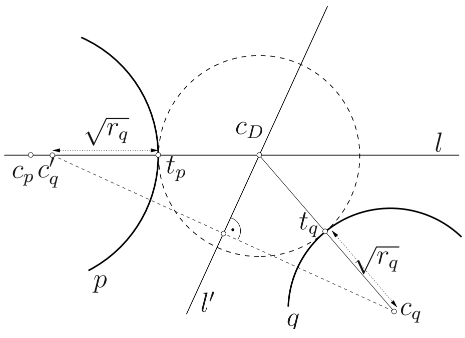


Fig. 1. Construction of a disk that is tangent to two arcs

touches p at t_p and is tangent to q (see Figure 1). First we construct the line l passing through the center c_p of the supporting circle of p and t_p. The center c_D of the maximal disk we are looking for has to be on l. As can be decided by the signs of curvature of p and q, one of the two points on l being at distance $\sqrt{r_q}$ (radius of the supporting circle of q) from t_p is denoted with c'_q. This point c'_q forms, together with c_D and c_q, an isosceles triangle. The bisector l' between c_q and c'_q also contains c_D, which means that c_D is the intersection point of l and l'. From all $q \neq p$ only one induces a disk (the smallest one) that lies completely inside S. This is the sought-after dividing disk D. The use of the ECAB structure guarantees certain algebraic properties of D. As the point c_p as well as t_p are rational, also the line l is rational. The value $\sqrt{r_q}$ is not rational in general, as a consequence so isn't c'_q. However, $c'_q \in \mathbb{Q}(\sqrt{r_q})^2$. Therefore also the point of intersection between l and l', being the center of D, is in this extension field. Values in $\mathbb{Q}(\sqrt{r_q})$ can be represented exactly by the rasqex numbers, which makes later point location on the bisector curves convenient (see Sections 5.2 and 6).

Furthermore, we note that the (rasqex) artificial arcs, stemming from the (rasqex) boundary circle of a dividing disks, always describe a local curvature maximum when used to extend the partial shapes. This is coherent with the definition of the ECAB-structure in Section 2.

5 Bisector Computation and Point Location

5.1 Bisector Computation

We next show how to compute the bisector between two rational arcs. Let C_p and C_q be the supporting circles of the two arcs with centers c_p and c_q and squared radii r_p and r_q, respectively:

$$C_p(x, y) := (x - (c_p)_x)^2 + (y - (c_p)_y)^2 - r_p$$
$$C_q(x, y) := (x - (c_q)_x)^2 + (y - (c_q)_y)^2 - r_q \ .$$

As before, we assume $c_p, c_q \in \mathbb{Q}^2$ and $r_p, r_q \in \mathbb{Q}$.

Definition 6. *The* bisector curve *between the two circles C_p and C_q consists of all points (x,y) in the plane for which*

$$\left||(x,y)-c_p| \pm \sqrt{r_p}\right| = \left||(x,y)-c_q| \pm \sqrt{r_q}\right| .$$

Roughly speaking, this bisector curve consists of all center points of circles, which share exactly one point (of tangency) with C_p and C_q respectively.

Theorem 1. *The bisector curve of the two circles C_p and C_q factors into two curves $B_1(x,y)=0$ and $B_2(x,y)=0$ with*

$$B_1(x,y) = (d_1^2+d_2^2-r^2)^2 - 4d_1^2d_2^2 \ \in\ \mathbb{Q}(\sqrt{r_pr_q})[x,y] \tag{3}$$
$$B_2(x,y) = (d_1^2+d_2^2-\tilde{r}^2)^2 - 4d_1^2d_2^2 \ \in\ \mathbb{Q}(\sqrt{r_pr_q})[x,y], \tag{4}$$

with $d_1 := d_1(x,y) := |(x,y)-c_p|$, $d_2 := d_2(x,y) := |(x,y)-c_q|$, $r := \sqrt{r_p}-\sqrt{r_q}$, and $\tilde{r} := \sqrt{r_p}+\sqrt{r_q}$.

Proof. For the bisector-curve there are two cases:

$$\textit{Case 1}\quad \left\{\begin{array}{l} \left\{\begin{array}{ll} & d_1+\sqrt{r_p} = d_2+\sqrt{r_q} \ \vee\ d_1-\sqrt{r_p} = d_2-\sqrt{r_q} \\ \vee & d_1+\sqrt{r_p} = -d_2+\sqrt{r_q} \vee d_1-\sqrt{r_p} = -d_2-\sqrt{r_q} \end{array}\right. \\ \Longleftrightarrow \left\{\begin{array}{ll} & d_1-d_2=-r \vee d_1-d_2=r \\ \vee & d_1+d_2=-r \vee d_1+d_2=r \end{array}\right. \qquad |\_^2 \\ \Longleftrightarrow d_1^2+d_2^2-r^2 \ = \ 2d_1d_2 \ \ \vee \ \ d_1^2+d_2^2-r^2 \ = \ -2d_1d_2 \qquad |\_^2 \\ \Longleftrightarrow (d_1^2+d_2^2-r^2)^2 \ = \ 4d_1^2d_2^2 \end{array}\right.$$

This is exactly the equation for $B_1(x,y)=0$. Similar for $B_2(x,y)=0$:

$$\textit{Case 2}\quad \left\{\begin{array}{l} \left\{\begin{array}{ll} & d_1+\sqrt{r_p} = -d_2-\sqrt{r_q} \vee d_1-\sqrt{r_p} = -d_2+\sqrt{r_q} \\ \vee & d_1+\sqrt{r_p} = d_2-\sqrt{r_q} \ \ \vee d_1-\sqrt{r_p} = d_2+\sqrt{r_q} \end{array}\right. \\ \Longleftrightarrow (d_1^2+d_2^2-\tilde{r}^2)^2 \ = \ 4d_1^2d_2^2 \end{array}\right.$$

Since $d_1^2, d_2^2 \in \mathbb{Q}[x,y]$, $\tilde{r}^2 = (\sqrt{r_p}+\sqrt{r_q})^2 = r_p + 2\sqrt{r_pr_q} + r_q \in \mathbb{Q}(\sqrt{r_pr_q})$, and $r^2 = (\sqrt{r_p}-\sqrt{r_q})^2 = r_p - 2\sqrt{r_pr_q} + r_q \in \mathbb{Q}(\sqrt{r_pr_q})$, it follows that $B_1(x,y), B_2(x,y) \in \mathbb{Q}(\sqrt{r_pr_q})[x,y]$. □

From now on B_1 and B_2 denote the curves described by $B_1(x,y)=0$ and $B_2(x,y)=0$ respectively.

Theorem 2. *B_1 and B_2 in Theorem 1 are conics, i.e., planar curves of degree two.*

Proof. We will prove that B_1 and B_2 are conics when the centers of the circles C_p and C_q lie on the x-axis and symmetrically on both sides of the y-axis:

$$C_p(x,y) := (x+d)^2+y^2-r_p, \quad C_q(x,y) := (x-d)^2+y^2-r_q \ .$$

This is no restriction because every pair of circles with d being half the distance between their two center points can be moved to this position by rotation and translation. B_1 and B_2 are then subject to the same transformation which does not change their degrees.

For C_p and C_q in this special position, we have

$$\begin{aligned} d_1^2 = d_1^2(x,y) &= \mid (x,y) - c_p \mid^2 = \mid (x,y) - (-d,0) \mid^2 = (x+d)^2 + y^2 \\ d_2^2 = d_2^2(x,y) &= \mid (x,y) - c_q \mid^2 = \mid (x,y) - (d,0) \mid^2 = (x-d)^2 + y^2 \ . \end{aligned}$$

This yields for the two cases

$$\textit{Case 1} \quad \begin{cases} \qquad (d_1^2 + d_2^2 - r^2)^2 = 4d_1^2 d_2^2 \\ \Longleftrightarrow (x^2 + d^2 + y^2 - \frac{r^2}{2})^2 = ((x+d)^2 + y^2)((x-d)^2 + y^2) \\ \Longleftrightarrow 0 = 4x^2d^2 - x^2r^2 - d^2r^2 - y^2r^2 + \frac{r^4}{4} \end{cases}$$

This is the quadratic equation for B_1.

$$\textit{Case 2} \quad \begin{cases} \qquad (d_1^2 + d_2^2 - \tilde{r}^2)^2 = 4d_1^2 d_2^2 \\ \Longleftrightarrow 0 = 4x^2d^2 - x^2\tilde{r}^2 - d^2\tilde{r}^2 - y^2\tilde{r}^2 + \frac{\tilde{r}^4}{4} \end{cases}$$

This is the quadratic equation for B_2. □

Altogether this means that the bisector of the two circles C_p and C_q in our original coordinate system factors into two conics over the field $Q_{pq} = \mathbb{Q}(\sqrt{r_p r_q})$, which is in rasqex.

Corollary 1. *Each of B_1 and B_2 is either a hyperbola or an ellipse or a pair of identical lines.*

Proof. Looking further at the equations for B_1 and B_2 in the special case where the center-points lie on the x-axis we first observe that B_1 and B_2 are the two conics described by

$$B_1(x,y) := bx^2 - ay^2 - ab, \qquad B_2(x,y) := \tilde{b}x^2 - \tilde{a}y^2 - \tilde{a}\tilde{b}$$

with

$$a = \frac{(\sqrt{r_p} - \sqrt{r_q})^2}{4} = \frac{r^2}{4}, \qquad b = d^2 - a = d^2 - \frac{r^2}{4}$$

and

$$\tilde{a} = \frac{(\sqrt{r_p} + \sqrt{r_q})^2}{4} = \frac{\tilde{r}^2}{4}, \qquad \tilde{b} = d^2 - \tilde{a} = d^2 - \frac{\tilde{r}^2}{4} \ .$$

First consider B_1. If $r_p = r_q$ we have $a = 0$, $b = d^2$ and $B_1(x,y) = d^2x^2$ consists of two identical lines along the y-axis. If $r_p \neq r_q$ it is true that $a > 0$ and

$$b > 0 \ \Leftrightarrow \ d^2 > \frac{r^2}{4} \ \Leftrightarrow \ 4d^2 > (\sqrt{r_p} - \sqrt{r_q})^2 \ \Leftrightarrow \ 2d > |\sqrt{r_p} - \sqrt{r_q}| \ .$$

That means,

– if $2d > |\sqrt{r_p} - \sqrt{r_q}|$, then $b > 0$ and B_1 is an hyperbola,
– if $2d = |\sqrt{r_p} - \sqrt{r_q}|$, then $b = 0$ and $B_1(x, y) = -ay^2$ consists of two identical lines along the x-axis,
– if $2d < |\sqrt{r_p} - \sqrt{r_q}|$, then $b < 0$ and B_1 is an ellipse.

For B_2 we always have $\tilde{a} > 0$ and

$$\tilde{b} > 0 \;\Leftrightarrow\; d^2 > \frac{\tilde{r}^2}{4} \;\Leftrightarrow\; 4d^2 > (\sqrt{r_p} + \sqrt{r_q})^2 \;\Leftrightarrow\; 2d > \sqrt{r_p} + \sqrt{r_q}\ .$$

– The two circles C_p and C_q do not intersect iff $2d > \sqrt{r_p} + \sqrt{r_q}$. In this case $\tilde{b} > 0$ and B_2 is a hyperbola.
– C_p and C_q touch tangentially iff $2d = \sqrt{r_p} + \sqrt{r_q}$. Then $B_2(x, y) = -\tilde{a}y^2$ consists of two identical lines along the x-axis.
– C_p and C_q intersect iff $2d < \sqrt{r_p} + \sqrt{r_q}$. In this case $\tilde{b} < 0$ and B_2 is an ellipse. □

5.2 Medial Axis Representation and Point Location

In order to compute and represent the medial axis of the exact circular arc boundary we must be able to analyze a bisector-conic over the extension field Q_{pq}. This means that in a so called *one-curve analysis* we will divide a bisector-conic B, described by $B(x, y) \in \mathbb{Q}(\sqrt{r_p r_q})[x, y] = Q_{pq}[x, y]$, into x-monotone arcs. This is not difficult and works analogously to the one-curve analysis of a conic over $\mathbb{Q}$ described in [3]. The bisector-conic B is split at its x-extreme points, that are points where $B(x, y)$ and the partial derivative $B(x, y)_y = \frac{\partial B(x,y)}{\partial y}$ vanish simultaneously. If the bisector-conic consists of a pair of identical lines, we make the defining polynomial square-free. Now every resulting x-monotone arc can be represented by a tuple $([le, ri], nr)$, where le and ri are the x-coordinates of the left and right endpoint, respectively. Since le and ri are roots of quadratic polynomials over $Q_{pq}[x]$, they can be represented by rasqex numbers. The branch number nr is either 0 or 1 and indicates which of the two x-monotone arcs of the curve above the x-interval $[le, ri]$ is meant.

As described in Section 4, one major step is point-location. For a given point $u = (u_x, u_y)$, the coordinates of which are rasqex, we have to determine the x-monotone arc of B_1 or B_2 it lies on. First of all we check whether u lies on B_1 or B_2 by testing

$$B_1(u_x, u_y) = 0 \quad \text{or} \quad B_2(u_x, u_y) = 0\ . \tag{5}$$

Since all the numbers in $B_1(u_x, u_y)$ and $B_2(u_x, u_y)$ are rasqex numbers, the exact test for zero can be realized by using the equality operator of the rasqex numbers. Let us assume that p lies on B_1. Next we use the $<$-operator of the rasqex numbers to determine the two x-monotone arcs of B_1 for which

$$le \le u_x \le ri\ . \tag{6}$$

The last step is to determine whether u lies on the upper or lower branch, i.e., algebraically whether u_y is the greater or smaller root of the polynomial $B_1(u_x, y)$. Since $B_1(u_x, y)$ is a quadratic polynomial the coefficients of which are rasqex, its two roots r_1 and r_2 can be computed symbolically by introducing a new square-root. Now we have to check whether

$$u_y - r_1 = 0 \quad \text{or} \quad u_y - r_2 = 0 \; . \tag{7}$$

Again this can be done by using rasqex numbers. Notice that in cases where locally around u neither the second bisector-conic nor the second arc pass by and all x-extreme points are far away, the three steps for point-location can be sped up by using isolating intervals for u_x and u_y and evaluating the expressions in (5), (6) and (7) with interval arithmetic, if desired.

5.3 Confining the Partial Axis

In our construction, the medial axis is computed as the union of bisector-conic segments. Each conic segment is limited by center points of artificial arcs. Consider the case where the bisector of two rational arcs on the circles C_p and C_q contribute to the medial axis, see for example Figure 3 case (b). The coordinates of the limiting center points of the artificial arcs are rasqex. With the algorithm described above the center points can be located on x-monotone arcs of the bisector-conic. If the bisector-conic is a line or hyperbola, the two center-points uniquely define the part of the medial axis we are interested in, possibly as a concatenation of x-monotone arcs. If the underlying bisector-conic is an ellipse, we have two possibilities for the partial axis. In this case we choose an additional rational point on one rational arc, say on C_p. With the algorithm from Section 4 we construct a third point on the bisector-conic. For the partial axis we choose the part of the bisector-curve between the two center points which contains this new point.

5.4 Center Points of Tritangent Circles

The center points of (at least) tritangent circles, being the branching points of the medial axis, are another kind of points which are needed for the confinement of the axis. We will show that the coordinates of such points are rasqex too, if the three defining circles are rational. A bisector curve between two rational circles is an algebraic curve of degree 4, and the branching point is one of the intersection points where all three bisectors between three circles meet.

There exist two different possibilities how a point on a bisector-curve may describe tangency at its footpoint on a defining circle.

Definition 7. *Consider a bisector-curve B and one of its two defining circles C. For a point $t \in B$ let t'_C be its unequivocal footpoint on C and Γ_C the open region bounded by C. Then we define the function $\varphi(t, C)$ on B as follows:*

$$\varphi(t, C) = \begin{cases} 0 & \textit{if} \quad \overline{t\, t'_C} \cap \Gamma_C = \emptyset \\ 1 & \textit{otherwise.} \end{cases}$$

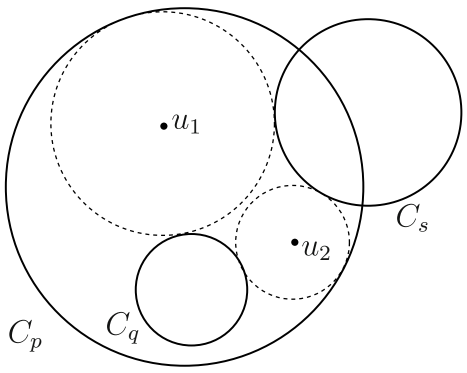


Fig. 2. Two tritangent circles resulting from one line of similitude of the Gergonne construction. $(\varphi(u_1, C_p), \varphi(u_1, C_q), \varphi(u_1, C_s)) = (\varphi(u_2, C_p), \varphi(u_2, C_q), \varphi(u_2, C_s)) = (1, 0, 0)$.

Roughly speaking $\varphi(t, C)$ is 0 if the circle with center t and radius $\overline{t\ t'_C}$ is "outer"-tangent to C, and 1 otherwise (see also Figure 2). As proved in Section 5 every bisector B consists of two conic curves, B_1 and B_2. By construction, the points of these two conics have certain properties concerning $\varphi(.,.)$ which we investigate next.

Lemma 1. *Consider the bisector B consisting of the two bisector-conics B_1 and B_2 and its two defining circles C_p and C_q, then*

$$\forall t \in B_1 \ : \ (\varphi(t, C_p), \varphi(t, C_q)) \in \{(0, 0), (1, 1)\} \tag{8}$$

$$\forall t \in B_2 \ : \ (\varphi(t, C_p), \varphi(t, C_q)) \in \{(0, 1), (1, 0)\} \ . \tag{9}$$

Proof. As derived in the proof of Theorem 1, for every point t on B_1 it holds that

$$\begin{aligned} & |t - c_p| + \sqrt{r_p} = |t - c_q| + \sqrt{r_q} \lor |t - c_p| - \sqrt{r_p} = |t - c_q| - \sqrt{r_q} \\ \lor \ \ & |t - c_p| + \sqrt{r_p} = -|t - c_q| + \sqrt{r_q} \lor |t - c_p| - \sqrt{r_p} = -|t - c_q| - \sqrt{r_q} \ . \end{aligned}$$

This leads to

$$\begin{aligned} & \varphi(t, C_p) = 1 \land \varphi(t, C_q) = 1 \lor \varphi(t, C_p) = 0 \land \varphi(t, C_q) = 0 \\ \lor \ \ & \varphi(t, C_p) = 1 \land \varphi(t, C_q) = 1 \lor \varphi(t, C_p) = 1 \land \varphi(t, C_q) = 1 \ . \end{aligned}$$

For every point x on B_2 it is

$$\begin{aligned} & |t - c_p| + \sqrt{r_p} = -|t - c_q| - \sqrt{r_q} \lor |t - c_p| - \sqrt{r_p} = -|t - c_q| + \sqrt{r_q} \\ \lor \ \ & |t - c_p| + \sqrt{r_p} = |t - c_q| - \sqrt{r_q} \lor |t - c_p| - \sqrt{r_p} = |t - c_q| + \sqrt{r_q} \ . \end{aligned}$$

This leads to

$$\begin{aligned} & \text{undefined} \lor \varphi(t, C_p) = 0/1 \land \varphi(t, C_q) = 1/0 \\ \lor \ \ & \varphi(t, C_p) = 1 \land \varphi(t, C_q) = 0 \lor \varphi(t, C_p) = 0 \land \varphi(t, C_q) = 1 \ . \end{aligned}$$

□

We are interested in the situation where three rational circles C_p, C_q and C_s are given. They define three bisectors: $B^{'}$ between C_p and C_q, $B^{''}$ between C_q and C_s and $B^{'''}$ between C_p and C_s. A branching point u, being the center of a tritangent circle, lies on all three bisectors and so $\varphi(u,.)$ is well defined for all three circles. Let

$$\Phi(u) := (\varphi(u, C_p), \varphi(u, C_q), \varphi(u, C_s)) \ . \tag{10}$$

Observation 1. *Depending on which bisector-conics intersect in a branching point u, we distinguish between four different sets of contact tuples. For all other possible combinations of three bisector-conics a common intersection point is impossible.*

$$u \in B_1^{'} \cap B_1^{''} \cap B_1^{'''} \quad \Rightarrow \quad \Phi(u) \in \{(0,0,0), (1,1,1)\} \tag{11}$$
$$u \in B_1^{'} \cap B_2^{''} \cap B_2^{'''} \quad \Rightarrow \quad \Phi(u) \in \{(0,0,1), (1,1,0)\} \tag{12}$$
$$u \in B_2^{'} \cap B_1^{''} \cap B_2^{'''} \quad \Rightarrow \quad \Phi(u) \in \{(1,0,0), (0,1,1)\} \tag{13}$$
$$u \in B_2^{'} \cap B_2^{''} \cap B_1^{'''} \quad \Rightarrow \quad \Phi(u) \in \{(1,0,1), (0,1,0)\} \tag{14}$$

For example, considering case (11), if $u \in B_1^{'} \cap B_1^{''} \cap B_1^{'''}$, then due to Lemma 1 it holds that

$$\begin{aligned} &(\varphi(u, C_p), \varphi(u, C_q)) \in \{(0,0), (1,1)\} \\ \wedge\ &(\varphi(u, C_q), \varphi(u, C_s)) \in \{(0,0), (1,1)\} \\ \wedge\ &(\varphi(u, C_p), \varphi(u, C_s)) \in \{(0,0), (1,1)\} \ . \end{aligned}$$

This is only true if $\varphi(u, C_p) = \varphi(u, C_q) = \varphi(u, C_s) = 0$ or $\varphi(u, C_p) = \varphi(u, C_q) = \varphi(u, C_s) = 1$. The other cases work analogously.

The construction of all possible circles that are tangent to three given circles is a much discussed topic, with various possible ways of solution (see e.g. [12]). It is folklore that there exist at most 8 different tritangent circles in this case. The Gergonne construction, named after french mathematician Joseph Diaz Gergonne, is based on inverse geometry and uses so called *lines of similitude*. For three circles in general position, there exist 4 lines of similitude. Each line induces at most 2 tritangent circles, which can both together be assigned to one specific case (11) to (14) from Observation 1. Note however, that e.g. for case (13) there may be two solutions of the form $(1, 0, 0)$ and none for $(0, 1, 1)$ (see Figure 2 for an example).

This means that constellations of three bisector-conics as shown in Observation 1 have at most two common intersection points. The x-coordinates of the intersection points of two of the three conics are roots of a degree four polynomial P_1 (which can be derived by a resultant computation). For another pair of conics we obtain another polynomial P_2. We now isolate the common x-components by computing the greatest common divisor $P' = \gcd(P_1, P_2)$. As at most two common solutions may exist, P' is a quadratic polynomial. Its roots can be represented exactly by rasqex numbers.[1] The same way the possibly two y-coordinates can be computed. This shows that the coordinates of the center points of tritangent circles can be represented as rasqex numbers and we get $2 \cdot 2 = 4$ candidates for them.

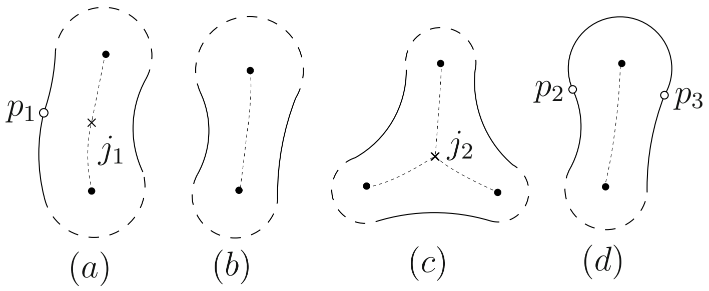


Fig. 3. The four combinatorial different base cases that may occur for the ECAB structure, as described in Section 6

6 Partial Axis Construction

In general, four combinatorially different base cases with ≤ 3 original arcs may stem from the iterative dividing process (Figure 3). The medial axes of these base cases are then represented directly by portions of algebraically simple circle bisectors. After the mathematical elaboration in Section 5 we now have a closer look at the combinatorial composition of the axes.

(a) The medial axis of base case (a) in Figure 3 consists of parts of the two bisectors between one of the two arcs incident to p_1 and the opposite arc. As we have a smooth transition at the rational point p_1, the two resulting bisector segments have a tangent point at the joint point j_1, which has rasqex coordinates. Together with the (rasqex) center points of the artificial arcs, j_1 is used to confine the required parts of the conic bisectors as described in Section 5.3.

(b) The axis is the segment of the two original arcs' bisector, which is confined by the two center points of the artificial arcs, see Section 5.3.

(c) The base case of this form represents the generic case for branching points of the medial axis. Its axis is composed of bisector parts from all pair constellations of original arcs. Let C_p, C_q and C_s be the three circles the original arcs lie on. For isolation of these segments, in addition to the three artificial center points, the intersection point j_2 has to be identified. How to compute

[1] In the special case where P_1 and P_2 have more than two common roots due to covertical intersection points, we shear the coordinate system, compute the center points of the tritangent circles in the sheared system and transform the result back to the original coordinate system.

the potential coordinates of such a point, which are also rasqex, is shown in Section 5.4. Finally we choose the correct intersection point among the computed ones by computing additional points on the bisector curves and following them starting at the center point of an artificial arc.

(d) Bisector construction is done as in case (b). However, unlike in case (b), one of the two confining points is not an artificial center point, but a center point of an original arc which represents a leaf point of the medial axis structure.

An arbitrary variation of the case depicted in Figure 4 may arise as a degenerate exception, which is an occurrence of an axis branching, where more than three bisectors meet in one point. For our dividing process this means that we arrive at a shape, whose boundary is an alternating sequence of artificial and original arcs. Here no generic dividing disk exists that would lead to combinatorially smaller partial shapes.

Granted algebraic correctness, as is the case in our setting, such degenerate cases can be detected easily: whenever an alternating arc sequence is recognized we compute the bisectors of all pairs of arcs which are only separated by one artificial arc. If all these bisectors intersect in one single point then a degenerate case has occured. Computation is based on the principle introduced in Section 5.4, meaning that again rasqex numbers are sufficient for exact calculation. This guarantees a correct indication of such a case which then can be handled accordingly. This elegant and intuitive handling of degnerate cases is one of the main improvements over the numerical afflicted approach in [2].

For the axis construction the bisector curves between original arcs that are neighbored via a single artificial arc are of interest. They all intersect in one common point which is, together with the center points of the artificial arcs, required for the segment confinement.

7 ECAB Construction

Our approach works on shapes S whose boundary ∂S is an ECAB. Thus in this final chapter we present a simple construction to obtain such a shape. In this construction only one single arc cannot be chosen rational but has to be rasqex. In accordance to the defintion of ECAB we shift this rasqex arc to a region where the curvature has a local maximum and therefore the medial axis has a leaf-point.

1. We start by choosing two rational points which represent the center of a circle and one endpoint of an arc on it (c_0 and p_0 in Figure 5). This circle is rational with respect to our definition in Section 2.
2. On the (rational) line through c_0 and p_0 we choose another rational point c_1, being the center of the next circle.
3. As c_1 and p_0 are rational, we can choose the next rational point p_1 in any ϵ-neighborhood around an arbitrary point on the circle with center c_1 and radius $\|c_1 - p_0\|$.

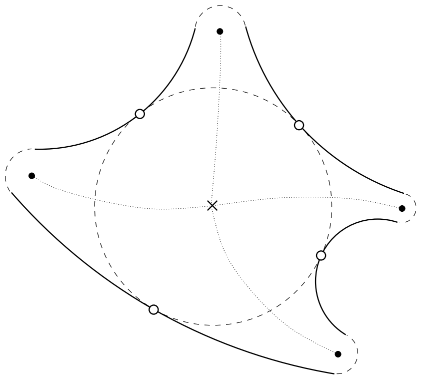


Fig. 4. Degenerate case example

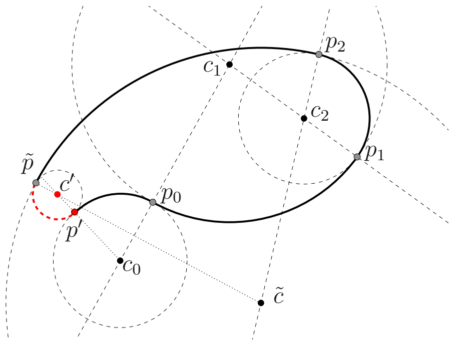


Fig. 5. Boundary construction which satisfies ECAB properties

4. We repeat the last two steps until we arrive at the closing circle which has to represent a local curvature maximum of the boundary.
5. It is in general not possible to find a rational closing circle. But following the construction of a maximal disk as described in Section 4 we obtain a rasqex arc with the supporting circle centered at c' and with radius $\|c' - p'\|$.

Note: If the closing arc resulting from this ECAB construction, which is generally not rational, is treated like an artificial arc, then it can be handled without any further modification as part of the base cases depicted in Figure 3.

8 Conclusion

We showed that, given a boundary essentially composed of rational arcs (ECAB), the Divide-&-Conquer approach for medial axis construction from [2] can be adapted for algebraically exact calculation. Furthermore, encouraged by [8], we were able to show that the rasqex number type is sufficient for all arising computations. Intermediate steps and procedures are discussed in detail, and a construction guide for a simple ECAB is provided. What is missing so far is an analysis of the degrees of the geometric predicates involved in our computation. This is left as a topic for future research.

An extension to circular boundaries with non-differentiable arc joints only causes an increase of base and bisector cases (see also [2]). The same applies for straight line segments, which also introduce parabolic curves to the axis. Exact computation for boundary representations with curves of algebraically higher degree may be a topic for future work, although the bisector complexity grows considerably in this context.

We would like to point out again, that the self-contained representation by rasqex numbers is a beneficial one. Correctness of the result and exactness during computation (allowing e.g. the efficient detection and handling of degenerate cases) are achieved by applying only moderate changes to the original (floating point) algorithm. We think that with the exact computability of the medial axis the algorithm recommends itself for implementation in geometric libraries as CGAL [1].

References

1. Cgal, Computational Geometry Algorithms Library, http://www.cgal.org
2. Aichholzer, O., Aigner, W., Aurenhammer, F., Hackl, T., Jüttler, B., Rabl, M.: Medial Axis Computation for Planar Free-Form Shapes. Computer Aided Design 41(5), 339–349 (2009)
3. Berberich, E., Eigenwillig, A., Hemmer, M., Hert, S., Mehlhorn, K., Schömer, E.: A Computational Basis for Conic Arcs and Boolean Operations on Conic Polygons. In: Möhring, R.H., Raman, R. (eds.) ESA 2002. LNCS, vol. 2461, pp. 174–186. Springer, Heidelberg (2002)
4. Blum, H.: A Transformation for Extracting New Descriptors of Shape. In: Wathen-Dunn, W. (ed.) Models for the Perception of Speech and Visual Form, pp. 362–380. MIT Press, Cambridge (1967)
5. Burnikel, C.: Rational Points on Circles. Research Report MPI-I-98-1-023. Max-Planck-Institut für Informatik, Saarbrücken (1998)
6. Burnikel, C., Funke, S., Mehlhorn, K., Schirra, S., Schmitt, S.: A Separation Bound for Real Algebraic Expressions. In: Meyer auf der Heide, F. (ed.) ESA 2001. LNCS, vol. 2161, pp. 154–265. Springer, Heidelberg (2001)
7. Elber, G., Kim, M.-S.: Bisector Curves of Planar Rational Curves. Computer-Aided Design 30(14), 1089–1096 (1998)
8. Emiris, I.Z., Karavelas, M.I.: The Predicates of the Apollonius Diagram: Algorithmic Analysis and Implementation. Computational Geometry 33(1-2), 18–57 (2006)
9. Emiris, I.Z., Tsigaridas, E.P., Tzoumas, G.M.: Exact Delaunay Graph of Smooth Convex Pseudo-circles: General Predicates, and Implementation for Ellipses. In: SIAM/ACM Joint Conference on Geometric and Physical Modeling 2009, pp. 211–222. ACM, New York (2009)
10. Emiris, I.Z., Tzoumas, G.M.: Exact and Efficient Evaluation of the Incircle Predicate for Parametric Ellipses and Smooth Convex Objects. Computer-Aided Design 40(6), 691–700 (2008)
11. Hanniel, I., Muthuganapathy, R., Elber, G., Kim, M.-S.: Precise Voronoi Cell Extraction of Free-form Rational Planar Closed Curves. In: SPM 2005, pp. 51–59. ACM, New York (2005)
12. Kunkel, P.: The Tangency Problem of Apollonius: Three Looks. BSHM Bulletin: Journal of the British Society for the History of Mathematics 22, 34–46 (2007)
13. Li, C., Yap, C., Pion, S., Du, Z., Sharma, V.: The Core Library, http://cs.nyu.edu/exact/core_pages/downloads.html
14. Mehlhorn, K., Näher, S.: The LEDA Platform for Combinatorial and Geometric Computing. Cambridge University Press, Cambridge (1999)
15. Schinzel, A., Sierpinski, W.: Elementary Theory of Numbers, 2nd edn. North-Holland Mathematical Library, vol. 31 (1988)
16. Yap, C.K., Dubé, T.: The Exact Computation Paradigm. In: Du, D.-Z., Hwang, F.K. (eds.) Computing in Euclidean Geometry, pp. 452–492. World Scientific Press, Singapore (1995)

Complex Bézier Curves and the Geometry of Polynomials

Rachid Ait-Haddou[1,2,⋆] and Taishin Nomura[1,2]

[1] The Center of Advanced Medical Engineering and Informatics, Osaka University, Japan
rachid@bpe.es.osaka-u.ac.jp

[2] Department of Mechanical Science and Bioengineering, Graduate School of Engineering Science, Osaka University, Japan

Abstract. In this paper, we study the shape of the control polygon of a complex Bézier curve over a complex interval. We show that the location of the complex roots of the polynomial dictates geometrical constraints on the shape of the control polygon. Along the work, new proofs and generalizations of the Walsh coincidence Theorem and the Grace Theorem are given. Applications of the geometry of the control polygon of complex polynomials to Bernstein type inequalities are discussed.

Keywords: Complex Bézier curves, complex de Casteljau algorithm, polar derivative, Grace Theorem, Walsh coincidence Theorem, Bernstein type inequalities.

1 Introduction

Parametric Bézier curves are widely used in Computer Aided Geometric Design due mainly to the practical ramifications of two fundamental properties. The first property relies on the existence of the notion of a control polygon, in which the shape of the parametric curve can be readily guessed or controlled. The second property appears in the so-called de Casteljau algorithm; a subdivision scheme for efficient polynomial evaluation and fast curve design [6]. To define the control polygon of a parametric Bézier curve, two polynomials with real coefficients P and Q are given, and upon taking the coefficients p_i (resp. q_i) of the polynomial P (resp. Q) with respect to the Bernstein basis over a real interval, we form the planar control polygon with vertices the points (p_i, q_i). In this work, we propose to study the notion of complex Bézier curves and associated control polygons [2]. In this setting, we start with a polynomial P with complex coefficients. Representing the complex polynomial P in the Bernstein basis over a complex interval $[a, b]$, we obtain complex coefficients p_i and therefore, a planar control polygon with vertices the points $(Re(p_i), Im(p_i))$. We generalize the de Casteljau algorithm for the evaluation of the polynomial at any complex argument and show that the location of the complex roots, or the critical points, of

⋆ Corresponding author.

the polynomial P dictates geometrical constraints on the shape of the control polygon. To illustrate what is meant by geometrical constraints, we can cite the following fact, proven in Section 4 : If a polynomial P of exact degree n has all its critical points on a circle C, then the control polygon of the polynomial P over any diameter of the circle C is a staircase polygon (Figure 1) with possibly some coincident control points. Some applications of the geometry of the control polygon to Bernstein type inequalities will be discussed. The paper is organized as follows : In the first section, we introduce the notion of complex Bézier curve and generalize the de Casteljau algorithm using the notion of polar form of a polynomial. For later use, we will also introduce the de Boor-Fix bracket on the linear space of polynomials. In Section 2, we introduce the notion of polar derivative. We give a new expression of the polar derivative that will enable us to give simple proofs and new generalizations to the well-known Laguerre Theorem, Grace Theorem and Walsh coincidence Theorem. The material of this section, did not stem only from our desire for a self-contained account of this work or from our need of new generalizations and simple proofs of the mentioned theorems, but it also stems from our dissatisfaction of some of the practices in the litterature in dealing with Laguerre Theorem, even in the classical book of Marden [11]. More precisely, the degree of a polynomial is explicit in the definition of the polar derivative and Laguerre Theorem is not always correct if we consider the polynomial under investigation as having degree higher than its exact degree. Therefore, naively, iterating Laguerre Theorem by successive use of the polar derivative is not correct in general, unless we check at each step the exact degree of the polynomial and apply the polar derivative accordingly. Our strategy to resolve this issue is to prove that Laguerre Theorem is true independently of the considered degree of the polynomial only if the chosen circular region contains the point at infinity. Using the inherent pseudo-symmetry of the de Boor-Fix bracket, we will be able to always choose circular regions that contain the point at infinity. This strategy allows for a clear proof and generalization of Grace Theorem and Walsh coincidence Theorem. In section 3, we study the geometry of the control polygon of a complex polynomial, when the roots of the polynomial lie in a circle, or in a disk or outside of a disk. In the last section, we apply the preceding informations on the geometry of the control polygon in deriving Bernstein, Turan and Erdos-Lax inequalities. The idea of using the polar form to derive these inequalities was already in the classical work of de Bruijn [5], in which there was not explicit terminology for the blossom. Our contribution in this section is merely to give elegant geometrical interpretations to these proofs and to stress the importance of studying the geometry of the control polygon of complex polynomials. The general connection between the geometry of the control polygon and Bernstein type inequalities can be understood as follows : Every condition on the control polygon is translated to a condition on the polar form, thereby leading to a Bernstein type inequality. Although we studied such connection in only three cases, namely a polynomial with roots in a circle, or inside a disk or outside a disk, we can ask similar questions about the shape of the control polygon for polynomials with roots in a half-plane, univalent

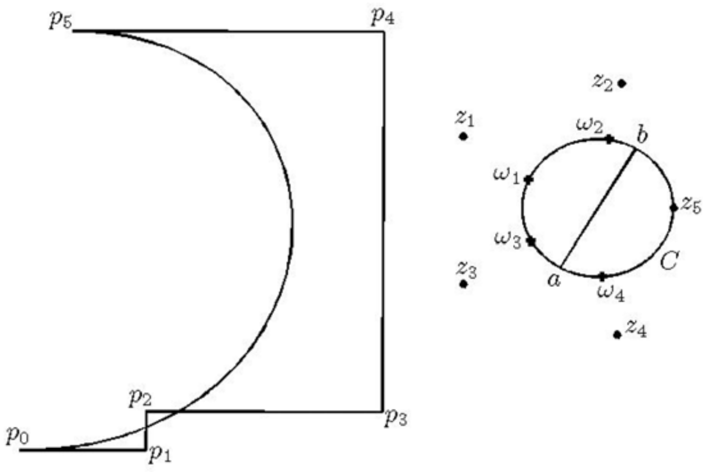


Fig. 1. For a polynomial with all critical points in a circle C, the control polygon over a diameter $[a, b]$ of the circle has a staircase shape. In the figure, p_i are the control points of the polynomial, z_i its roots and ω_i its critical points.

polynomials in a disk, reciprocal polynomials,... Each of this learned properties on the control polygon will be associated to a Bernstein type inequality. Due to space limitation, we were not able to touch upon these questions and it will be the theme of a subsequent effort on the subject.

2 Complex Bézier Curves and Blossom

2.1 Complex Bézier Curves and Complex de Casteljau Algorithm

Within this work, we will always refer to polynomials of degree at most n as polynomials of degree n. We will often stress the exact degree of the polynomial when it is needed.

Let P be a complex polynomial of degree n, then for any complex interval $[a, b]$,$(a \neq b)$, we can express the polynomial P in the Bernstein basis over the interval $[a, b]$ as

$$P(z) = \sum_{i=0}^{n} p_i B_i^n(z), \tag{1}$$

where the Bernstein polynomials B_i^n, $i = 0, ..., n$ are given by

$$B_i^n(z) = C_i^n \left(\frac{b-z}{b-a}\right)^{n-i} \left(\frac{z-a}{b-a}\right)^{i}, \qquad C_i^n = \frac{n!}{i!(n-i)!}.$$

Under the identification of a complex number $z = x + iy$ with the planar point $z = (x, y)$, we obtain from (1), a polygon $(p_0, p_1, ..., p_n)$ called the control polygon associated with the polynomial P over the interval $[a, b]$. A fundamental concept for the rest of this work, in dealing with the control polygon of Bézier curves, is the notion of *polar form* (or blossom) [13] associated with the polynomial P.

Definition 1. *Let P be a polynomial of degree n. There exists a unique multi-affine, symmetric function in n variables $f_P \colon \mathbb{C}^n \longrightarrow \mathbb{C}$ such that for each z in $\mathbb{C}$ we have $f_P(z, z, \ldots., z) = P(z)$. The function f_P is called the polar form or the blossom associated with the polynomial P.*

The control polygon $(q_0, q_1, ..., q_n)$ of a polynomial P over any interval $[c, d]$, $(c \neq d)$ can be computed using the polar form f_P of the polynomial P as follows:

$$q_i = f_P(c^{\{n-i\}}, d^{\{i\}}), \quad i = 0, 1, \ldots, n, \tag{2}$$

in which the notation $z^{\{k\}}$ indicates that the complex number z has to be repeated k times.

To generalize the de Casteljau algorithm to complex polynomials, it is convenient to introduce the notion of shape parameter of a triangle as in [10].

Definition 2. *Let $T = (a, b, c)$ be an oriented triangle in the complex plane. We define the shape Δ_T of the triangle as*

$$\Delta_T = \frac{a-b}{a-c}.$$

It is straightforward to show that two oriented triangles T_1 and T_2 have the same shape if and only if they are similar. Using this notion of shape, we can show the following

Lemma 1. *Let P be a polynomial of degree to n, let f_P be its polar form and $u_1, u_2, \ldots, u_{n-1}$ be complex numbers. Then for any complex numbers z_1, z_2, and z_3, the two oriented triangles $T_1 = (f_P(u_1, u_2, \ldots, u_{n-1}, z_1), f_P(u_1, u_2, \ldots, u_{n-1}, z_2), f_P(u_1, u_2, \ldots, u_{n-1}, z_3))$ and $T_2 = (z_1, z_2, z_3)$ are similar.*

Proof. The shape of the triangle T_1 is

$$\Delta_{T_1} = \frac{f_P(u_1, u_2, \ldots, u_{n-1}, z_1) - f_P(u_1, u_2, \ldots, u_{n-1}, z_2)}{f_P(u_1, u_2, \ldots, u_{n-1}, z_1) - f_P(u_1, u_2, \ldots, u_{n-1}, z_3)}. \tag{3}$$

By the multi-affinity of the polar form, we have

$$f_P(u_1, \ldots, u_{n-1}, z_1) - f_P(u_1, \ldots, u_{n-1}, z_2) = \frac{1}{n}(z_1 - z_2) f_{P'}(u_1, \ldots, u_{n-1}),$$

where P' is the derivative of the polynomial P. Inserting the last equation into (3) leads to $\Delta_{T_1} = (z_1 - z_2)/(z_1 - z_3)$, which is the shape of the triangle T_2. □

The multi-affinity of the blossom reveals that by an appropriate iterative use of Lemma 1, over a control polygon of a polynomial over a complex interval, we can evaluate the value of the polynomial at complex arguments - as well as the value of its polar form over arbitrary poles - in a similar fashion as in the case of parametric Bézier curves. Namely, a complex de Casteljau algorithm in which instead of using the multi-affinity over a line, we use the triangle similarity over the complex plane. The idea of the complex de Casteljau algorithm is easily illustrated using two simple examples. In the first example, we compute the value of the polynomial at a complex argument, while in the second example, we use the complex de Casteljau algorithm for computing the value of the blossom of the polynomial over certain complex poles.

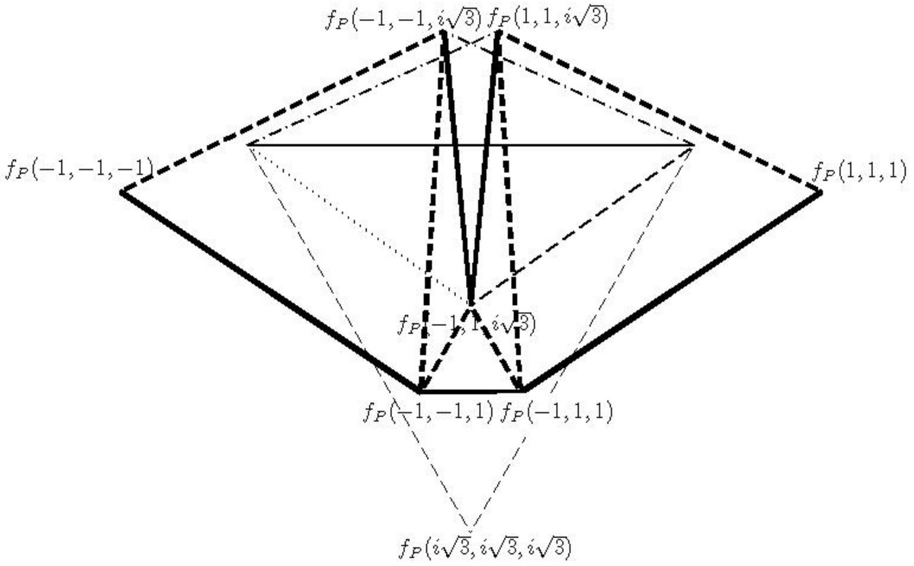
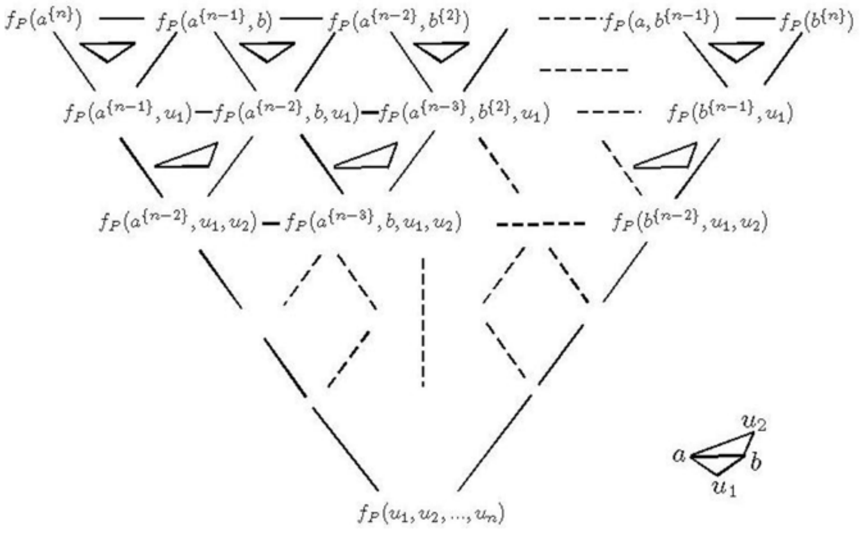


Fig. 2. The generalized de Casteljau algorithm for the evaluation of a complex polynomial at a complex argument. Refer to Example 1 for a detailed explanation of the Figure.

Example 1. Consider the cubic polynomial $P(z) = z^3 + 3iz^2 + 6z + 5$ and consider the control polygon $\mathcal{P}$ of the polynomial P over the interval $[-1, 1]$. From the polar form, we can calculate the control polygon of the polynomial P as $(p_0, p_1, p_2, p_3) = (-2 + 3i, 4 - i, 6 - i, 12 + 3i)$ (Figure 2). In order, for example, to calculate the value $P(\sqrt{3}i)$ we proceed as follows : Since the oriented triangle $(-1, 1, \sqrt{3}i)$ is equilateral, the point $f_P(-1, -1, \sqrt{3}i)$ is such that the oriented triangle $(p_0, p_1, f_P(-1, -1, \sqrt{3}i))$ is also equilateral. The same argument can be applied to the points $f_P(-1, 1, \sqrt{3}i), f_P(1, 1, \sqrt{3}i)$. This is the level one complex de Casteljau algorithm. To proceed, we calculate the point $f_P(-1, \sqrt{3}i, \sqrt{3}i)$ as the point such that the oriented triangle $(f_P(-1, -1, \sqrt{3}i), f_P(-1, 1, \sqrt{3}i),$ $f_P(-1, \sqrt{3}\ i, \sqrt{3}i))$ is equilateral, and so on until we reach the last level of the complex de Casteljau algorithm where we calculate $f_P(\sqrt{3}i, \sqrt{3}i, \sqrt{3}i) = P(\sqrt{3}i)$. (See Figure 2).

Example 2. Consider the same cubic polynomial $P(z) = z^3 + 3iz^2 + 6z + 5$ in which we would like to compute the polar form $f_P(1/2, u_1, u_2)$, where u_1 and u_2 are, for example, points in the complex plane such that the triangle $(-1, u_1, 1)$ is isosceles with apex angle at u_1 equal to $\pi/2$, and the triangle $(-1, u_2, 1)$ is also isosceles with an apex angle of $\pi/4$ at u_2. To proceed, we first insert the pole $1/2$ into the blossom of the polynomial P using only the multi-affinity of the blossom. This is the first level de Casteljau algorithm that would give us the points $q_1 = f_P(-1, -1, 1/2)$, $q_2 = f_P(-1, 1, 1/2)$ and $q_3 = f_P(1, 1, 1/2)$. To insert the pole u_1 into the polar form, we proceed by drawing on each edge of the polygon (q_1, q_2, q_3) isosceles triangles with apex angle $\pi/2$. This second level de Castejau algorithm gives us an edge $[f_P(-1, 1/2, u_1), f_P(1, 1/2, u_1)]$. Drawing an isosceles triangle based on this edge with an apex angle equal to $\pi/4$ leads to the desired polar value $f_P(1/2, u_1, u_2)$.

We can represent the complex de Casteljau algorithm using the familiar triangular array (Figure 3), in which every triangle inside the triangular polar values represent the similarity constraint within the construction.



$f_P(a^{\{n\}})$
$f_P(a^{\{n-1\}}, b)$
$f_P(a^{\{n-2\}}, b^{\{2\}})$
$f_P(a, b^{\{n-1\}})$
$f_P(b^{\{n\}})$
$f_P(a^{\{n-1\}}, u_1)$
$f_P(a^{\{n-2\}}, b, u_1)$
$f_P(a^{\{n-3\}}, b^{\{2\}}, u_1)$
$f_P(b^{\{n-1\}}, u_1)$
$f_P(a^{\{n-2\}}, u_1, u_2)$
$f_P(a^{\{n-3\}}, b, u_1, u_2)$
$f_P(b^{\{n-2\}}, u_1, u_2)$
$f_P(u_1, u_2, \ldots, u_n)$
a
b
u_1
u_2

Fig. 3. The generalized de Casteljau algorithm for the computation of the polar form at complex polar values. The triangle inside each two given polar values and the computed one reflect the similarity constraint that has to be respected in inserting the polar value. For example, the value $f_P(a^{\{n-1\}}, u_1)$ is computed from the given polar values $f_P(a^{\{n\}})$ and $f_P(a^{\{n-1\}}, b)$ as the point such that the triangle (in the figure) inside these three values is similar to the oriented triangle $(f_P(a^{\{n\}}), f_P(a^{\{n-1\}}, b), f_P(a^{\{n-1\}}, u_1))$. Refer to Example 2 for an illustration.

2.2 de Boor-Fix Bracket and Blossom

In the linear space $\mathcal{P}_n$ of polynomials of degree n, we define the de Boor-Fix bracket operator $[,]_n$ as follows : Given two polynomials P, Q of degree n, $[P,Q]_n$ is given by

$$[P,Q]_n = \sum_{k=0}^{n} \frac{(-1)^{n-k}}{n!} P^{(k)}(z) Q^{(n-k)}(z),$$

where $R^{(j)}$ stand for the j-th derivative of a polynomial R.

The de Boor-Fix operator $[,]_n$ is bilinear and independent of z. Moreover, it is symmetric if n is an even integer and skew-symmetric if n is an odd integer. There is an ultimate relationship between the de Boor-Fix operator and the polar form, and it can be stated as follows [7] : For any complex numbers $u_1, u_2, ..., u_n$ and any polynomial P of degree n, we have

$$[P, (u_1 - z)(u_2 - z) \ldots (u_n - z)]_n = f_P(u_1, u_2, \ldots, u_n),$$

where f_P is the polar form of the polynomial P. If the polynomials P and Q have exact degree n, with respective roots $z_1, z_2, ..., z_n$ and $u_1, u_2, ..., u_n$, we have

$$[P,Q]_n = \frac{(-1)^n Q^{(n)}(z)}{n!} f_P(u_1, u_2, ..., u_n). \tag{4}$$

However, if the polynomial Q has an exact degree $m \leq n$, we have

$$[P,Q]_n = \frac{(-1)^m m!}{n!} Q^{(m)}(z) [P^{(n-m)}, Q]_m. \tag{5}$$

Note that if we write the polynomials P and Q explicitly as

$$P(z) = \sum_{k=0}^{n} C_n^k a_k z^k, \quad Q(z) = \sum_{k=0}^{n} C_n^k b_k z^k,$$

then we have

$$[P,Q]_n = \sum_{k=0}^{n} (-1)^{n-k} C_n^k a_k b_{n-k}.$$

The last equation leads to the familiar definition of apolar polynomials

Definition 3. *Two polynomials P and Q of degree n are said to be apolar if, and only if*

$$[P,Q]_n = 0.$$

3 Polar Derivative of a Polynomial

Let P be a complex polynomial of degree n and ζ be a complex number. The polynomial

$$P'_\zeta(z) = f_P(\zeta, z, z, \ldots, z)$$

is called the polar derivative of the polynomial P with respect to the pole ζ. It can be expressed explicitly as

$$P'_\zeta(z) = P(z) + \frac{(\zeta - z)}{n} P'(z).$$

In the following, we will give a different and convenient expression of the polar derivative that enables us to study the location of its zeros in the complex plane. We consider the following two linear operators in the $\mathbb{C}$-space of polynomials $d : \mathcal{P}_n \to \mathcal{P}_{n-1}$ and $\psi_{\zeta,n} : \mathcal{P}_n \to \mathcal{P}_n$ defined by

$$d(P(z)) = P'(z) \quad \text{and} \quad \psi_{\zeta,n}(P(z)) = (z-\zeta)^n \, P(\zeta + \frac{1}{z-\zeta})\,. \tag{6}$$

Introducing the map

$$\hat{\zeta} : z \mapsto \hat{\zeta}(z) = \zeta + 1/(z-\zeta), \tag{7}$$

we have $\hat{\zeta} \circ \hat{\zeta}(z) = z$ for all $z \neq \zeta$, from which we deduce

$$\psi_{\zeta,n} \circ \psi_{\zeta,n} = I_d\,, \tag{8}$$

where I_d is the identity on $\mathcal{P}_n$. By direct inspection, we can show the following proposition relating the polar derivative of a polynomial P with respect to a pole ζ and the two operators d and $\psi_{\zeta,n}$ [1].

Proposition 1. *For every polynomial P of degree n and for any complex number ζ, we have*

$$\psi_{\zeta,n-1} \circ d \circ \psi_{\zeta,n}(P) = n\, P'_\zeta. \tag{9}$$

Before studying the location of the zeros of the polar derivative, we introduce the familiar notion of circular regions. A *circular region* of the complex plane is defined as the image of either the closed or the open unit disk under a nonsingular Möbius map γ of the form

$$\gamma(z) = \frac{az+b}{cz+d},$$

where a, b, c and d are complex numbers such that $ad - bc \neq 0$. Möbius maps are $1-1$ mapping of the extended plane into itself with the property of mapping every circle onto either a circle or a line, and every line onto either a circle or a line. Therefore, a circular region is one of the following : an open disk, a closed disk, an open half plane, a closed half plane including ∞, the open exterior of a circle including ∞ or a closed exterior of a circle including ∞. We will always refer to the circular regions that include ∞, as simply, the circular regions containing the point at infinity.

Now, consider a polynomial P of exact degree n with all its roots z_i, $i = 1, ..., n$, in a circular region C and let ζ be a complex number outside of C. The map $\hat{\zeta}$ in equation (7) sends the complex point ζ to ∞, thus it maps the circular region C to a circular region $\hat{\xi}(C)$ not containing the point at infinity.

Thereby, $\hat{\xi}(C)$ is necessarily convex. Therefore, the roots $\hat{\zeta}(z_i)$ of the polynomial $\psi_{\zeta,n}(P)$ of equation (6) belong to $\hat{\zeta}(C)$. By Gauss-Lucas Theorem, the roots of the polynomial $d \circ \psi_{\zeta,n}(P)$ also belong to $\hat{\zeta}(C)$. Applying again the involutive transformation $\hat{\zeta}$ to the roots of the polynomial $d \circ \psi_{\zeta,n}(P)$ according to equation (9), we conclude that the roots of the polar derivative P'_ζ belong to the circular region C. Summarizing,

Theorem 1. *(***Laguerre Theorem***). If all the zeros of a polynomial P of exact degree n lie in a circular region C and if ζ is a complex number outside of C, then the zeros of the polynomial P'_ζ belong to the circular region C.*

Laguerre Theorem is wrong if we do not assume that the polynomial P is of exact degree n. For example, consider the polynomial $P(z) = z^2 + 1$ viewed as a polynomial of degree $n > 2$. The polar derivative of the polynomial P with respect to a pole ζ is given by $nP_\zeta(z) = (n-2)z^2 + 2\xi z + n$. The roots i and $-i$ of the polynomial P belong to the circular region C given as the closed unit disk. However, for the pole $\zeta = \sqrt{n(n-2)}$ lying outside the circular region C, the polar derivative has a root $-\zeta/(n-2)$ of multiplicity 2, which for $n > 2$ lies outside the circular region. To understand what fails in the proof of Laguerre Theorem in such a case, we could first give the following informal explanation: The polynomial P above, when considered as a polynomial of degree $n > 2$, can be viewed as a polynomial with two roots in the circular region C and $n-2$ roots at ∞. Therefore, not all the roots of the polynomial P lie in the circular region C as the point ∞ does not belong to the closed unit disk. However, to understand what exactly fails within the proof, we proceed as follows : When a polynomial P of exact degree s and roots $z_1, z_2, ..., z_s$ in a circular region C, is viewed as a polynomial of degree n, we rewrite the transformation $\psi_{\zeta,n}(P)$ as

$$\psi_{\zeta,n}(P(z)) = (z-\zeta)^{(n-s)}\big((z-\zeta)^s\, P(\zeta + \frac{1}{z-\zeta})\big).$$

In this case, the polynomial $\psi_{\zeta,n}(P)$ has as roots the complex numbers $\hat{\zeta}(z_i)$ and ζ as a root of multiplicity $(n-s)$. If ζ does not belong to the circular region $\hat{\zeta}(C)$, we cannot use the essence of Gauss-Lucas Theorem. However, if we assume that the circular region C contains the point at infinity, and the pole ζ does not belong to C, then ζ belongs to the circular region $\hat{\zeta}(C)$, otherwise, by applying again the transformation $\hat{\zeta}$ to both $\hat{\zeta}(C)$ and ζ, we arrive at the contradiction that there is a separating line between the circular region C and the point at infinity. In this case, Laguerre Theorem remains true independently of the viewed degree of the polynomial. Therefore, we have

Proposition 2. *If all the zeros of a polynomial P of degree n lie in a circular region C that contains the point at infinity and if ζ is a complex number outside of C, then the zeros of the polynomial P'_ζ belong to the circular region C.*

Let P be a polynomial of degree n. As we have already seen, the polar derivative P'_{ζ_1} of the polynomial P viewed as a polynomial of degree n can be written as $f_P(\zeta_1, z^{\{n-1\}})$. If we consider the polar derivative of the polynomial

$Q = P'_{\zeta_1}$, viewed as a polynomial of degree $n-1$, with respect to a pole ζ_2, then we have $Q'_{\zeta_2}(z) = f_P(\zeta_1, \zeta_2, z^{\{n-2\}})$. Therefore, for any complex numbers $\zeta_1, \zeta_2, ..., \zeta_s, (s \leq n)$, the polynomial $f_P(\zeta_1, \zeta_2, ..., \zeta_s, z^{\{n-s\}})$ is the successive polar derivative of the polynomial P with respect to the poles $\zeta_1, \zeta_2, ..., \zeta_s$ and in which at each iterative step, we view the obtained polynomial as a unit degree less than the degree of the polynomial obtained in the preceding iterative step. It is very important to respect this rule on the degree at each iterative step when performing the polar derivative, even if at a certain step the degree of the polynomial is strictly less than expected, as the following example shows :

Example 3. Consider the polynomial $P(z) = z^3 - 3z^2 + 6z + 1$. The polar derivative of the polynomial P with respect to the pole $\zeta = 1$ is $P'_1(z) = 2z + 3$, which is of exact degree less than the expected number 2. In order to express the polynomial $f_P(1, i, z)$ as successive polar derivative, we should consider the polynomial P'_1 as a polynomial of degree 2 and proceed with its polar derivative with respect to the pole i. Otherwise, if we consider the polynomial P'_1 as a polynomial of degree 1 and proceed with the polar derivative, we would not find the right answer, namely $f_P(1, i, z)$.

As proposition 2 is independent of the exact degree of the considered polynomial, we have

Theorem 2. *Let P be a polynomial of degree n whose roots lie in a circular region C that contains the point at infinity. Let $\zeta_1, \zeta_2, ..., \zeta_k, k \leq n$ be k complex numbers outside of C. Then, if the polynomial $Q(z) = f_P(\zeta_1, \zeta_2, ..., \zeta_k, z^{\{n-k\}})$ is not constant, all its roots lie in the circular region C.*

This Theorem leads us to the well-known Grace Theorem [12]

Theorem 3. *(***Grace Theorem***). If P and Q are two apolar polynomials of exact degree n and if one of them has all its roots in a circular region C, then the other will have at least one zero in C.*

Proof. Let $z_1, z_2, ..., z_n$ be the roots of the polynomial P and $u_1, u_2, ..., u_n$ the roots of the polynomial Q . From the hypothesis of apolarity, and by (4), we have $f_P(u_1, u_2, ..., u_n) = 0$ and $f_Q(z_1, z_2, ..., z_n) = 0$. Let us assume that the roots of the polynomial Q are inside a circular region C, while the roots of the polynomial P are outside the circular region C, which is also a circular region which we denote by D. As the two regions C and D are complementary, one of them will contain the point at infinity. Without loss of generality, we can assume that it is the region D that contains the point at infinity and that $u_i \neq u_j$ for $i \neq j$. By Theorem 2, the polynomial $f_P(u_1, u_2, ..., u_{i-1}, u_{i+1}, ..., u_n, z)$, if not constant, has a root in the circular region D, but by hypothesis it has also a root in the region C, namely u_i. Therefore, by the multi-affinity of the blossom, we have $f_P(u_1, u_2, ..., u_{i-1}, u_{i+1}, ..., u_n, z) \equiv 0$, for $i = 1, ..., n$. In the special case in which the polynomial $f_P(u_1, u_2, ..., u_{i-1}, u_{i+1}, ..., u_n, z)$ is constant, then the constant has to be zero as we have $f_P(u_1, u_2, ..., u_n) = 0$. Therefore, in both cases, we have $f_P(u_1, u_2, ..., u_{i-1}, u_{i+1}, ..., u_n, z) \equiv 0$. That shows, again by the multi-affinity of the blossom, that the polynomial $P \equiv 0$, which leads to a contradiction. Therefore, one of the roots of the polynomial P belong to the circular region C. □

Grace's Theorem leads to the following celebrated Walsh coincidence Theorem [11].

Theorem 4. *(***Walsh coincidence Theorem***). Let f be a symmetric multi-affine function of n complex variables and total degree equal to n . Let $u_1, u_2, \ldots, u_n$ be n complex numbers which lie in a circular region C. Then, there exists a ζ in C such that*

$$f(u_1, u_2, \ldots, u_n) = f(\zeta, \zeta, \ldots, \zeta).$$

Proof. Let P be the polynomial defined by $P(z) = f(z, z, ..., z)$. Then the polynomial P is of exact degree n and $f = f_P$ its blossom. Consider the polynomial of exact degree n defined by $Q(z) = P(z) - f_P(u_1, u_2, ..., u_n)$. From the hypothesis on the complex numbers u_i, we have $f_Q(u_1, u_2, ..., u_n) = 0$. Therefore, by Grace Theorem, the polynomial Q has a root in the circular region C. □

Consider, now, a polynomial P of exact degree n and roots $z_i, i = 1, ..., n$ and consider another polynomial Q of exact degree $m \leq n$ such that $[P, Q]_n = 0$. In this case, we have $f_Q(z_1, z_2, ..., z_n) = 0$ where f_Q is the blossom of Q viewed as a polynomial of degree n. Therefore, from (5), we have $f_Q(w_1, w_2, ..., w_m) = 0$ where $w_1, ..., w_m$ are the roots of the polynomial $P^{(n-m)}$. Consequently, by Grace Theorem, any circular region that contains the roots of the $(n-m)$-derivative of the polynomial P contains a root of the polynomial Q. Therefore, we get the following generalization of Grace and Walsh Theorems.

Theorem 5. *Let P be a polynomial of exact degree n and Q a polynomial of exact degree $m \leq n$. If the two polynomials P and Q are apolar, then any circular region containing the roots of $P^{(n-m)}$, contains at least one root of Q.*

Equivalently, we can formulate this corollary in term of the blossom as

Corollary 1. *Let f be a symmetric multi-affine function of n complex variables and total degree $m \leq n$. Let $u_1, u_2, \ldots, u_n$ be n complex numbers which lie in a circular region C. Then, there exists a ζ in a circular region containing the roots of $P^{(n-m)}$, where P is the polynomial $P(z) = \prod_{i=1}^{n}(z - u_i)$ such that*

$$f(u_1, u_2, \ldots, u_n) = f(\zeta, \zeta, \ldots, \zeta).$$

The last corollary essentially expresses the fact that the control polygon of the degree elevation has the potential of giving more refined informations on the location of the roots of the polynomial. This is consistent with the fact that the control polygon of degree elevation reveals more information on the shape of the curve and even converges to the curve with successive degree elevation.

Note that if in Theorem 5, the roots of the polynomial P are in a disk or a half plane (closed or open), then the roots of the successive derivatives are also located in the same region. Therefore, Theorem 5, constitutes a generalization of the following results of Aziz [4] and Jain [9].

Corollary 2. *Let P be a polynomial of exact degree n and Q a degree $m \leq n$ polynomial such that P and Q are apolar. If the roots of P are in a disk D (resp. a half plane H), then at least one of the roots of the polynomial Q lies in the disk D (resp. the half plane H).*

Equivalently, we can formulate this corollary in term of the blossom as

Corollary 3. *Let f be a symmetric multi-affine function of n complex variables and total degree $m \leq n$. Let $u_1, u_2, \ldots, u_n$ be n complex numbers which lie in a disk D (resp. a half plane H). Then there exists a ζ in D (resp. H) such that*

$$f(u_1, u_2, \ldots, u_n) = f(\zeta, \zeta, \ldots, \zeta).$$

4 Geometry of the Control Polygon

In this section, we study the shape of the control polygon of a complex polynomial when the relative location of the roots of the polynomial with respect to a disk is given.

4.1 Polynomial with Roots on a Circle

Proposition 3. *Let P be a polynomial of exact degree n whose roots lie in a circle C and let ζ be a complex number that lies in the circle C. Then all the roots of the polar derivative P'_ζ lie in the circle C, unless $P(z) = \lambda(z - \zeta)^n$ and in which case we have $P'_\zeta \equiv 0$.*

Proof. The proof follows the same lines as in our proof of Laguerre Theorem, in which this time we use Rolle Theorem instead of Gauss-Lucas Theorem. Let $z_1, z_2, ..., z_n$ be the roots of the polynomial P which are assumed to lie in a circle C and let ζ be a complex number in C. Let us assume that s $(0 \leq s < n)$ roots $z_1, z_2, ..., z_s$ of the polynomial P coincide with the pole ζ. In this case, the polynomial $\psi_{\zeta,n}(P)$ is of exact degree $n - s$, where $\psi_{\zeta,n}$ is defined in (6). As the map $\hat{\zeta}$ of (7) sends the point ζ to infinity, the points $\hat{\zeta}(z_i), i = s + 1, ..., n$ belong to a line L. Therefore, the roots of the polynomial $\psi_{\zeta,n}(P)$ belong to L, thereby by Rolle Theorem, the roots of the polynomial $d \circ \psi_{\zeta,n}(P)$ also belong to L. By applying again the transformation $\hat{\zeta}$ with respect to (9), we arrive at the statement that the polar derivative P'_ζ has $n-s-1$ roots on the circle C and $s - 1$ roots that coincide with the pole ζ. Therefore, all the roots of the polar derivative P'_ζ lie in the circle C. The only case we did not yet deal with is the one in which $s = n$, i.e.; all the roots of the polynomial P coincide with the pole ζ. In this case, we have $P(z) = \lambda(z - \zeta)^n$ and then $P'_\zeta \equiv 0$. □

Consider now a polynomial P of exact degree n whose roots $z_i, i = 1, ..., n$ lie in a circle C, we further assume that the polynomial P is not of the form $P(z) = \lambda(z-\rho)^n$. Let ζ_1 be a complex number in the circle C. From the last Proposition, the polynomial $f_P(\zeta_1, z, z, ..., z)$ is a polynomial of degree $n-1$ with all its roots in the circle C. To be able to iterate the statement of Proposition 3, we have to show that the polynomial $f_P(\zeta_1, z, z, ..., z)$ is of exact degree $n-1$. Writing the polynomial P in the monomial basis as $P(z) = a_n z^n + a_{n-1}z^{n-1} + ... + a_0$ and computing the higher term of the polar derivative, we find $f_P(\zeta_1, z, z, ..., z) = \big(a_n\zeta_1 + (a_{n-1}/n)\big)z^{n-1} + ...$ Therefore, the polynomial P_{ζ_1} is of degree strictly less than $n-1$ if and only if the pole ζ_1 satisfy the equation $\zeta_1 = (\sum z_i)/n$. As the roots z_i belong to the circle C, its centroid lie strictly inside the circle unless all the roots coincide and in which case we have $P(z) = \lambda(z-\zeta_1)^n$. Therefore, we have shown that the polynomial $f_P(\zeta_1, z, z, ..., z)$ is of exact degree $n-1$. Iterating then the process of taking successive polar derivatives and applying Proposition 3, we find that for any complex numbers $\zeta_1, \zeta_2, ..., \zeta_k$ in the circle C, the polynomial $f_P(\zeta_1, ..., \zeta_k, z, ...z)$ has all its roots in the circle C. In particular, consider two complex numbers a, b such that the segment $[a, b]$ is a diameter of the circle C, then for any $i = 1, ..., n$, the degree 1 polynomial $Q_i(z) = f_P(a^{n-i}, b^{i-1}, z)$ has its unique root in the circle C, for every $i = 1, ..., n-1$, i.e; there exist ρ_i in the circle C such that $f_P(a^{n-i}, b^{i-1}, \rho_i) = 0$. Since the oriented triangle $(f_P(a^{n-i}, b^{i-1}, a), f_P(a^{n-i}, b^{i-1}, b), f_P(a^{n-i}, b^{i-1}, \rho_i))$ is similar to the oriented triangle (a, b, ρ_i), we deduce that the control polygon of the polynomial P over the interval $[a, b]$ satisfy $< p_i, p_{i+1} >= 0$ for $i = 0, ..., n-1$.(the notation $< z_1, z_2 >$ is referred here to the scalar product of the complex numbers z_1 and z_2 i.e., $Re(z_1\bar{z_2})$). Consider now the case in which the polynomial P is of the form $P(z) = \lambda(z-\rho)^n$, where ρ lies in the circle C and let $[a, b]$ be a diameter of the circle C. If ρ is different from a and b, then the preceding arguments are still valid and the control polygon $(p_0, p_1, ..., p_n)$ of the polynomial P satisfy $< p_i, p_{i+1} >= 0$ for $i = 0, ..., n-1$. In the case the complex number ρ coincide with a (resp. b) then the control polygon $(p_0, p_1, ..., p_n)$ is $(0, 0, ..., 0, \lambda(b-\rho)^n)$ (resp. $(\lambda(a-\rho)^n, 0, 0, ..., 0)$) and therefore, in all cases we have $< p_i, p_{i+1} >= 0$ for $i = 1, ..., n-1$. Therefore

Theorem 6. *If all the zeros of a polynomial P of exact degree n lie in a circle C, then its control polygon $(p_0, p_1, ..., p_n)$ over a diameter $[a, b]$ of the circle C satifies $< p_i, p_{i+1} >= 0$ for $i = 0, ..., n-1$ i.e.; for every $i = 0, ..., n-1$ the circle of diameter $[p_i, p_{i+1}]$ passes through the origin (Figure 4).*

In Figure 4, we have shown an example of the control polygon of a polynomial with all its roots in a circle C over a diameter of the circle. We can notice the existence of two orthogonal lines L_1 and L_2 that intersect at the origin and in which the location of the control points of the polynomial alternates between the two lines L_1 and L_2. In the case all the control points are non-zeros, such a property can be deduced directly from Theorem 6. However, in the case where some control points are zero, the existence of the two orthogonal lines is not obvious.

In the following, we will show that the condition of Theorem 6 imposes the existence of such two orthogonal lines or in other words, we will show that we also have $\det(p_i, p_{i+2}) = 0$ for $i = 1, ...n-2$. To do so, we first notice that for a polynomial with exact degree n with all its roots in a circle, a sequence of control points, over a diameter of the circle, of the form $(p_0, p_1, ..., p_k, 0, 0, ..., 0, p_{k+s}, ..., p_n)$ in

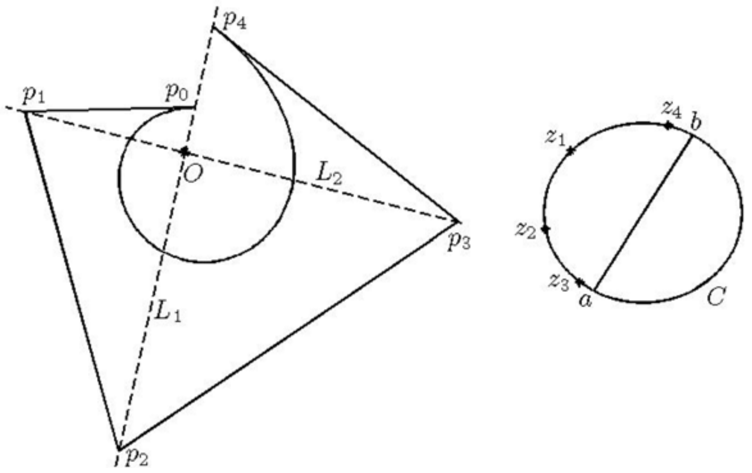


Fig. 4. For a polynomial with all it roots in a circle C, there exist two orthogonal lines L_1 and L_2 intersecting at the origin, such that the control points, over a diameter $[a, b]$ of the circle, alternate between the two lines. In the figure, p_i are the control points of the polynomial, O the origin and z_i the roots of the polynomial.

which the number of zero control points between p_k and p_{k+s} is strictly bigger than 1 ($s > 2$) and $p_k \neq 0$ and $p_{k+s} \neq 0$ is forbiden. Indeed, if such a sequence exists, then the polynomial $Q(z) = f_P(a^{\{n-(k+s)+1\}}, b^{\{k+1\}}, z^{\{s-2\}}) \equiv 0$. However, from the proof of Theorem 6, such a polynomial has exact degree $s-2 > 0$ unless $P(z) = \lambda(z-a)^n$ (resp. $\lambda(z-b)^n$) and in which case the control point $p_k = 0$ (resp. $p_{k+s} = 0$), contradicting our initial assumption. Therefore, the only non obvious case to consider is a sequence of control points of the form $(p_0, p_1, ..., p_k, 0, p_{k+2}, ..., p_n)$, where $p_k \neq 0$ and $p_{k+2} \neq 0$. To such a sequence, we associate the polynomial $Q(z) = f_P(a^{\{n-k-2\}}, b^{\{k\}}, z, z)$. The polynomial Q has all its roots in the circle C and has $(p_k, 0, p_{k+2})$ as a control polygon over the interval $[a, b]$. Let c be the point in the circle C such that the diameter [a,b] is orthogonal to the segment $[(a + b)/2, c]$ i.e., $c = (a + ib)/(1 + i)$. The polar derivative Q'_c of the polynomial Q with respect to the pole c has its only root in the circle C. The control points of Q'_c over the interval $[a, b]$ are given by $q_0 = p_k/(1 + i)$ and $q_1 = ip_{k+2}/(1 + i)$. Therefore, we have $< q_0, q_1 >= 0$, which can be rewritten as $< p_k, ip_{k+2} >= -det(p_k, p_{k+1}) = 0$. Therefore, we have proven

Corollary 4. *If all the zeros of a polynomial P of exact degree n lie in a circle C, then its control polygon $(p_0, p_1, ..., p_n)$ over a diameter $[a, b]$ of the circle C satifies $< p_i, p_{i+1} >= 0$ for $i = 0, ..., n-1$ and $det(p_i, p_{i+2}) = 0$ for $i = 0, ..., n-2$ i.e.; there exist two orthogonal lines L_1 and L_2 that intersect at the origin and in which the location of the control points p_i alternates between the two lines.*

From now on, a control polygon that satisfy the geometrical conditions of the last corollary will be called an *orthogonal control polygon.*

Remark 1. Consider a polynomial P with real coefficients and of exact degree n. Let us assume that all the roots of the polynomial P lie in the unit circle. Let $(p_0, p_1, ..., p_n)$ be the control polygon of P over the interval $[-1, 1]$. In this case, all the control points are reals. Moreover, the control points should alternate between two orthogonal lines. The only possible scenario for this to happen is that the odd or the even control points of the polynomial are all equal to zero.

In the following, we show that if the control polygon of a polynomial P is orthogonal over a diameter $[a, b]$ of a circle C, then the control polygon of P over a different diameter remains orthogonal, independently of the condition on the roots of the polynomial. In some sense, it shows that it is redundant to always stress that the orthogonality is true for any diameter of the circle, if it is true for a single diameter. For such purpose, we need the following lemma.

In the following, we show that if the control polygon of a polynomial P is orthogonal over a diameter $[a, b]$ of a circle C, then the control polygon of P over a different diameter remains orthogonal, independently of the condition on the roots of the polynomial. In some sense, it shows that it is redundant to always stress that the orthogonality is true for any diameter of the circle, if it is true for a single diameter. For such purpose, we need the following lemma.

Lemma 2. *Let P be a polynomial of exact degree $n \geq 3$ such that its control polygon $(p_0, p_1, ..., p_n)$ over a diameter $[a, b]$ of a circle C is orthogonal. Let ζ be a complex number in C, then the polynomial P'_ζ has the property that its control polygon $(q_0, q_1, ..., q_{n-1})$ over $[a, b]$ is orthogonal.*

Proof. Let $(p_0, ..., p_n)$ be the control polygon of the polynomial P over $[a, b]$ and $(q_0, q_1, ..., q_{n-1})$ be the control polygon of P'_ζ over the same interval. We have

$$q_i = \frac{b-\zeta}{b-a} p_i + \frac{\zeta - a}{b-a} p_{i+1}.$$

Using the fact that $< p_i, p_{i+1} >= 0$, we have

$$< q_i, q_{i+1} >= \frac{1}{\|b-a\|^2} \big(< (b-\zeta)p_i, (\zeta - a)p_{i+2} > + \|p_{i+1}\|^2 < \zeta - a, b - \zeta > \big).$$

Since $[a, b]$ is a diameter of the circle C and ζ lies in C, we have $< \zeta - a, b - \zeta >= 0$. Moreover, using the fact that there exist real numbers λ_i such that $p_{i+2} = \lambda_i p_i$, leads to $< q_i, q_{i+1} >= 0$, for $i = 0, ..., n-2$. Similar computations show that $det(q_i, q_{i+2}) = 0$. $\square$

Corollary 5. *Let P be a polynomial of exact degree n such that its control polygon $(p_0, p_1, ..., p_n)$ over a diameter $[a, b]$ of a circle C is orthogonal. Then, the control polygon $(q_0, q_1, ..., q_n)$ of the polynomial P over a different diameter $[c, d]$ is also orthogonal.*

Proof. Let us assume that the degree of the polynomial P is greater than 3. Let $[c, d]$ be a diameter of the circle C and $(q_0, q_1, ..., q_n)$ be the control polygon of

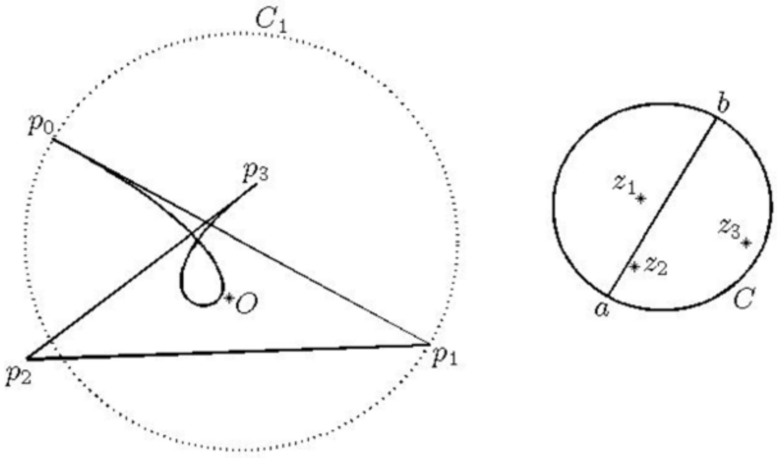


Fig. 5. For a polynomial with all its roots inside a disk with boundary the circle C, and a control polygon over a diameter $[a, b]$ of the circle, the disk of diameter any two consecutive control points contains the origin. In the figure, p_i are the control points of the polynomial, O the origin and z_i the roots of the polynomial.

the polynomial P over the diameter $[c, d]$. Using iteratively Lemma 2, with the hypothesis that the control polygon $(p_0, p_1, ..., p_n)$ of the polynomial P over the interval $[a, b]$ is orthogonal, we know that for any $0 \leq j \leq n-2$, the control polygon of the polynomial $Q_j(z) = f_P(c^{\{n-2-j\}}, d^{\{j\}}, z, z)$ over the interval $[a, b]$ is orthogonal. The control polygon of the polynomial Q_j over the interval $[a, b]$ is given by $r_0 = f_P(c^{\{n-2-j\}}, d^{\{j\}}, a, a)$, $r_1 = f_P(c^{\{n-2-j\}}, d^{\{j\}}, a, b)$ and $r_2 = f_P(c^{\{n-2-j\}}, d^{\{j\}}, b, b)$. Therefore, we have

$$(b-a)^2 q_j = r_0(b-c)^2 + 2r_1(c-a)(b-c) + r_2(c-a)^2,$$

$$(b-a)^2 q_{j+1} = r_0(b-c)(b-d) + r_1((c-a)(b-d) + (d-a)(b-c)) + r_2(c-a)(d-a),$$

and

$$(b-a)^2 q_{j+2} = r_0(b-d)^2 + 2r_1(d-a)(b-d) + r_2(d-a)^2.$$

Straightforward computations then show that $< q_j, q_{j+1} >= 0$ and $\det(q_j, q_{j+2}) = 0$ (It may be helpful, and without loss of generality, to assume that C is the unit circle and take $a = e^{i\theta}, b = -e^{i\theta}, c = e^{i\phi}$ and $d = -e^{i\phi}$ in carrying the computations). The case of degree 1 polynomials is obvious, while for degree 2 polynomials, we can use direclty the last equations to describe the control polygon of the quadratic polynomial over the interval $[c, d]$. $\square$

If we apply Theorem 6, to the derivative of the polynomial P, we get an interesting geometrical property of the control polygon of polynomials with critical points in a circle, namely

Proposition 4. *If a polynomial P of exact degree n has all its critical points in a circle C, then the control polygon $(p_0, p_1, ..., p_n)$ of the polynomial P over a diameter $[a, b]$ of the circle C satisfies $< p_{i+1} - p_i, p_{i+2} - p_{i+1} >= 0$, for $i = 0, ..., n-2$. i.e.; the control polygon has a staircase shape (Figure 1).*

4.2 Polynomial with Roots Inside a Disk

Consider a polynomial P of exact degree n with all its roots inside a closed disk D and let ζ_1 be a complex number in the circle ∂D. Then, similar arguments that was used in Laguerre Theorem proof, show that the polynomial $f_P(\zeta_1, z, z, ..., z)$ is a polynomial of degree $n-1$ with all its roots inside the disk D. (Note that we can also invoke Hurwitz Theorem on the continuity of the roots with respect to the coefficients of the polynomial. As the roots are inside the disk, no root can escape to infinity during the continuity process). Moreover, similar arguments as in the proof of Theorem 6, shows that the polynomial $f_P(\zeta_1, z, z, ..., z)$ is of exact degree $n-1$, unless the polynomial P is of the form $\lambda(z-\zeta_1)^n$. Excluding this particular case, allows us to iterate this process for any complex numbers $\zeta_1, \zeta_2, ..., \zeta_k$ in the circle ∂D, i.e.; the polynomial $f_P(\zeta_1, ..., \zeta_k, z, ...z)$ has all its roots in the disk D. In particular, consider two complex numbers a, b such that the segment $[a, b]$ is a diameter of the circle ∂D, then the degree 1 polynomial $Q_i(z) = f_P(a^{n-i}, b^{i-1}, z)$ has it unique root in the disk D, i.e; there exists ζ in the disk D such that $f_P(a^{n-i}, b^{i-1}, \zeta) = 0$. Since the oriented triangle $(f_P(a^{n-i}, b^{i-1}, a), f_P(a^{n-i}, b^{i-1}, b), f_P(a^{n-i}, b^{i-1}, \zeta))$ is similar to the oriented triangle (a, b, ζ), we have $< p_i, p_{i+1} > \leq 0$. The case in which the polynomial P is of the form $\lambda(z-\rho)^n$ has been already treated in the last section, and in all cases we have the following

Theorem 7. *If all the zeros of a polynomial P of exact degree n lie inside a disk D, then its control polygon $(p_0, p_1, ..., p_n)$ over a diameter $[a, b]$ of the circle ∂D satifies $< p_i, p_{i+1} > \leq 0$ for $i = 0, ..., n-1$ i.e.; for every $i = 0, ..., n-1$ the disk of diameter $[p_i, p_{i+1}]$ contains the origin (Figure 5).*

Remark 2. Unlike the orthogonality condition on the control polygon of polynomials with roots in a circle, the geometrical condition on the control polygon for polynomials with all roots inside the disk depends on the interval in which the control polygon is taken. A simple illustration of this point would be the quadratic polynomial $P(z) = z^2 - 2iz$. Its control polygon over the interval $[-1, 1]$ is $(p_0, p_1, p_2) = (1 + 2i, -1, 1 - 2i)$ and satisfies $< p_i, p_{i+1} > \leq 0, i = 0, 1$. However, its control polygon $(q_0, q_1, q_2) = (-3, 1, 1)$ over the interval $[-i, i]$ does not satisfy the condition $< q_1, q_2 > \leq 0$

If we apply Theorem 7 to the derivative of the polynomial P, we get the following interesting geometrical property of the control polygon of polynomials with critical points inside a disk, namely

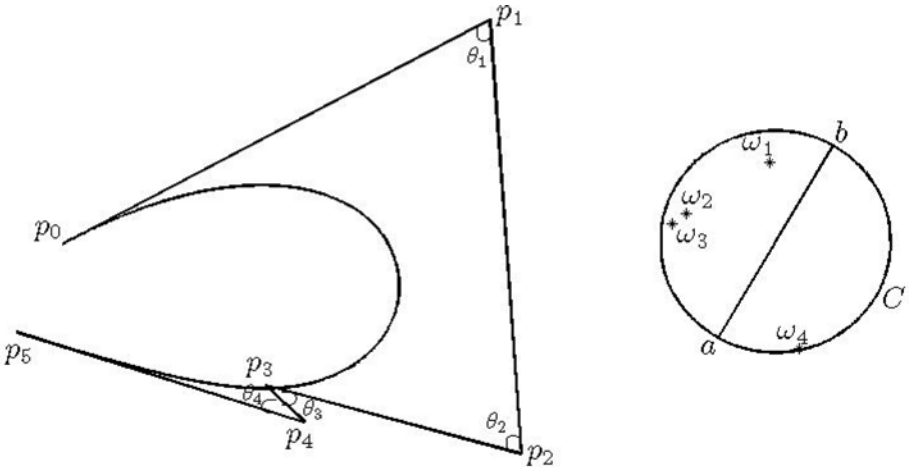


Fig. 6. For a polynomial with all its critical points inside a disk with boundary the circle C, and a control polygon over a diameter $[a, b]$ of the circle, the non-oriented apex angle θ_i of the triangle (p_{i-1}, p_i, p_{i+1}) at the vertex p_i is less or equal to $\pi/2$. In the figure, p_i are the control points of the polynomial, ω_i its critical points.

Proposition 5. *If a polynomial P of exact degree n has all its critical points inside a disk D, then the control polygon $(p_0, p_1, ..., p_n)$ of the polynomial P over a diameter $[a, b]$ of the circle ∂D satisfies $< p_{i+1} - p_i, p_{i+2} - p_{i+1} > \leq 0$, for $i = 0, ..., n-2$. i.e.; the non-oriented angle of the triangle (p_i, p_{i+1}, p_{i+2}) at the vertex p_{i+1} is smaller than $\pi/2$ (Figure 6).*

4.3 Polynomial with Roots Outside a Disk

Consider a polynomial P with all its roots outside a disk D. As the roots of P belong to a circular region that contains the point at infinity, the issue of the exact degree of the polynomial does not manifest itself. Therefore, using the same proof as in Laguerre Theorem, complemented with a similar treatement on the control polygon as in the last section leads to

Theorem 8. *If all the zeros of a polynomial P of degree n lie outside a disk D, then the control polygon $(p_0, p_1, ..., p_n)$ over a diameter $[a, b]$ of the circle ∂D satifies $< p_i, p_{i+1} > \geq 0$ for $i = 0, ..., n-1$ i.e.; for every $i = 0, ..., n-1$ the disk of diameter $[p_i, p_{i+1}]$ does not contain the origin (Figure 7).*

Again, applying this result to the derivative of the polynomial P leads to

Proposition 6. *If a polynomial P of degree n has all its critical points outside a disk D, then the control polygon $(p_0, p_1, ..., p_n)$ of the polynomial P over a diameter $[a, b]$ of the circle ∂D satisfies $< p_{i+1} - p_i, p_{i+2} - p_{i+1} > \geq 0$, for $i = 0, ..., n-2$. i.e.; the non-oriented angle of the triangle (p_i, p_{i+1}, p_{i+2}) at the vertex p_{i+1} is larger than $\pi/2$ (Figure 8).*

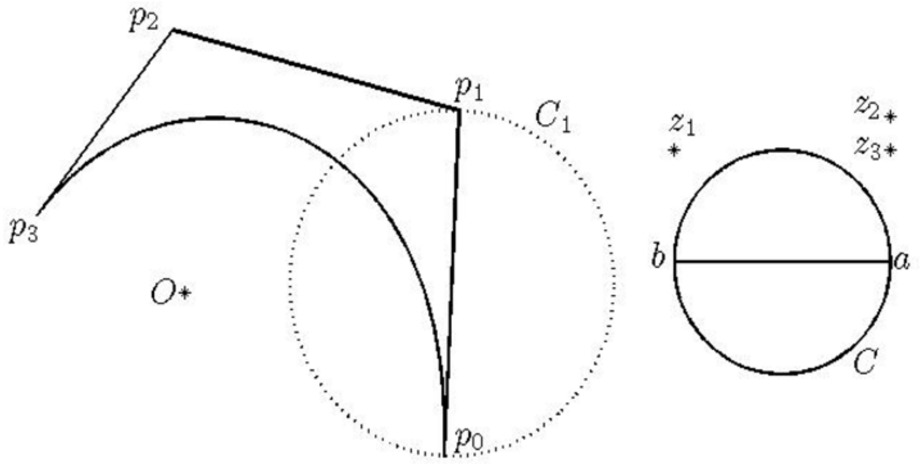


Fig. 7. For a polynomial with all its roots outside a disk with boundary the circle C, and a control polygon over a diameter $[a, b]$ of the circle, the disk of diameter any two consecutive control points does not contain the origin. In the figure, p_i are the control points of the polynomial, O the origin and z_i the roots of the polynomial.

5 Bernstein-Type Inequalities

In this section, we will show some applications of the shape of the control polygon of a polynomial to Bernstein type inequalities.

For a complex polynomial P, and for a region D in the complex plan, we denote

$$||P||_D = max_{z \in D}|P(z)|.$$

Theorem 9. *(***Bernstein Inequality***). Let P be a complex polynomial of degree n and D a disk of radius R. Then*

$$||P'||_D \leq \frac{n}{R}||P||_D.$$

Proof. Let f_P be the polar form of the polynomial P viewed as a polynomial of degree n. By Corollary 3, we have $|f_P(u_1, u_2, ..., u_n)| \leq ||P||_D$ for all $u_1, u_2, .., u_n$ in the disk D, otherwise, we will have a complex number ζ in the disk D such that $|f_P(u_1, u_2, ..., u_n)| = |P(\zeta)| > ||P||_D$. In particular, for any complex number a in ∂D, we have $|f_P(a, a, .., a)| \leq ||P||_D$ and $|f_P(a, a, ..a, 2c - a)| \leq ||P(z)||_D$; c being the center of the disk D ($[a, 2c - a]$ is a diameter of the disk D). Therefore

$$|f_P(a, a, .., a) - f_P(a, a, ..a, 2c - a)| \leq 2||P||_D.$$

By the multi-affinity of the polar form, the last equation can be rewritten as

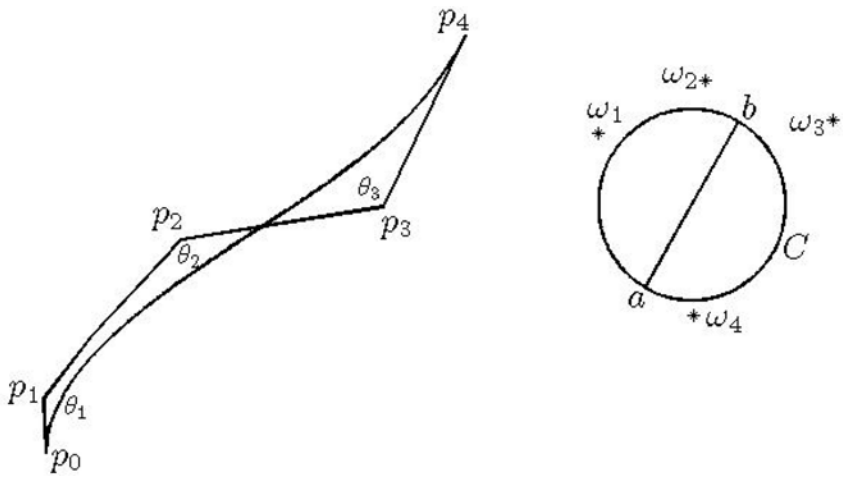


Fig. 8. For a polynomial with all its critical points outside a disk with boundary the circle C, and a control polygon over a diameter $[a, b]$ of the circle, the non-oriented apex angle θ_i of the triangle (p_{i-1}, p_i, p_{i+1}) at the vertex p_i is greater or equal to $\pi/2$. In the figure, p_i are the control points of the polynomial, ω_i its critical points.

$$|P'(a)| \leq \frac{n}{|a-c|}||P||_D = \frac{n}{R}||P||_D.$$

As the last inequality is true for any a in ∂D, the maximum principle leads to a proof of the Bernstein inequality. □

Theorem 10. *(***Turan Inequality***). Let P be a complex polynomial of exact degree n, with all its roots in a disk D of radius R. Then*

$$||P'||_D \geq \frac{n}{2R}||P||_D. \tag{10}$$

Proof. Let f_P be the polar form of the polynomial P. Let a be an arbitrary complex number in ∂D. Since all the roots of the polynomial P are inside the disk D, by Theorem 7, the disk of diameter the first two control points $f_P(a, a, ..., a)$ and $f_P(a, a, ..., a, 2c-a)$ of the polynomial over the diameter $[a, 2c-a]$ contains the origin (Figure 9). Therefore,

$$|f_P(a, a, ..., a)| \leq |f_P(a, a, ..., a) - f_P(a, a, ..., a, 2c-a)|.$$

Using the multi-affinity of the polar form, the last equation can be rewritten as

$$|f_P(a, a, ..., a)| \leq \frac{2|a-c|}{n}|P'(a)| \leq \frac{2R}{n}||P'||_D.$$

As a is arbitrary in ∂D, the maximum principle leads to Turan inequality. □

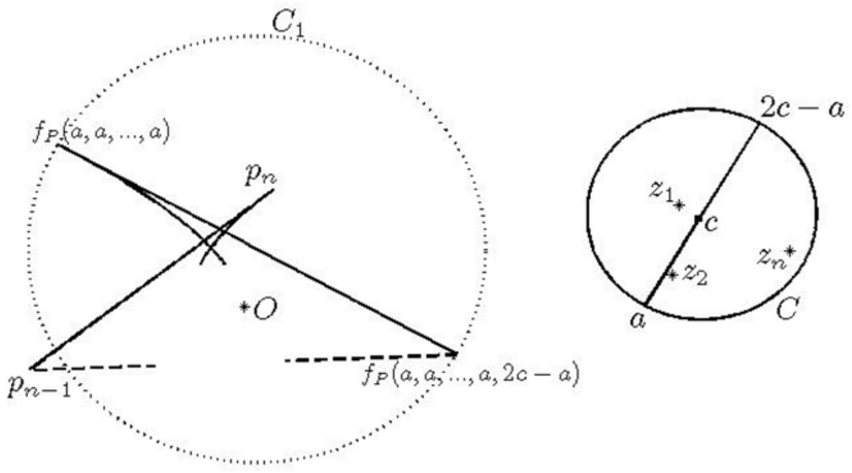


Fig. 9. A geometric illustration of the proof of Turan Inequality. Refer to the proof for an explanation.

Theorem 11. (**Lax-Erdos Inequality**). *Let P be a complex polynomial of degree n, with no roots in a disk D of radius R. Then*

$$||P'||_D \le \frac{n}{2R}||P||_D. \tag{11}$$

Proof. Let f_P be the polar form of the polynomial P and let a be an arbitrary complex number in ∂D. Since all the roots of the polynomial P are outside the disk D, then, by Theorem 8, the disk $\mathcal{H}$ with boundary C_1 of diameter the two first control points $f_P(a,a,...,a)$ and $f_P(a,a,...,a,2c-a)$ does not contain the origin (Figure 10). Therefore,

$$|(f_P(a,a,...,a)+f_P(a,a,...,a,2c-a))/2| \ge |(f_P(a,a,...,a)-f_P(a,a,...,a,2c-a))/2|.$$

Using the multi-affinity of the polar form, the last equation can be rewritten as

$$|f_P(a,a,...,a,c)| \ge \frac{|a-c|}{n}|P'(a)|.$$

Consider the point A in the circle C_1 defined as the farthest intersection of the line $[0, f_P(a,a,...,a,c)]$ and the circle $\partial\mathcal{H}$. The point A can be expressed as $A = f_P(a,a,...,a,\delta)$ with δ in the circle C (Figure 10). We have

$$|f_P(a,a,...,a,\delta)| = |f_P(a,a,...,a,c)| + \frac{|a-c|}{n}|P'(a)|.$$

Therefore,

$$||P||_D \ge |f_P(a,a,...,a,\delta)| \ge \frac{2|a-c|}{n}|P'(a)| = \frac{2R}{n}|P'(a)|.$$

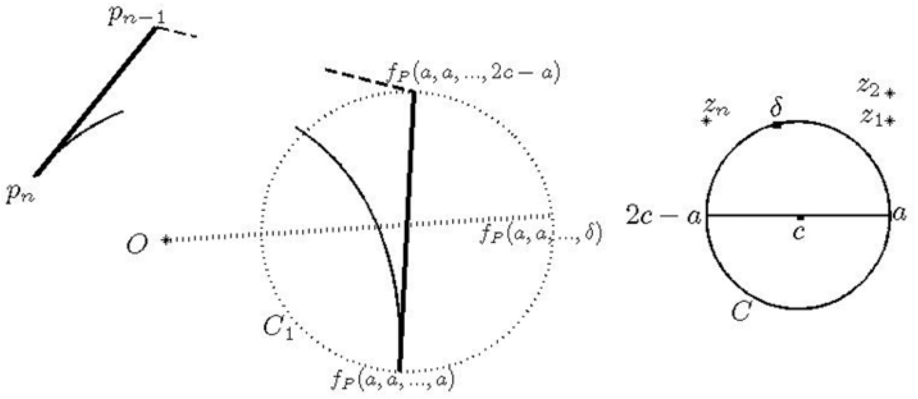


Fig. 10. A geometric illustration of the proof of Lax-Erdos Inequality. Refer to the proof for an explanation.

Again, as a is arbitrary in ∂D, the maximum principle leads to the proof of the Erdos-Lax inequality. □

6 Conclusions

In this work, we initiated the study of the geometry of the control polygon of a complex Bézier curve. We showed that the location of the roots or the critical points of the polynomial dictates geometrical constraints on the shape of the control polygon. Some applications to Bernstein type inequalities were given. Although we have studied the geometry of the control polygon for three special cases, namely, when the roots of the polynomial are in a circle, or belong to a disk or lie outside a disk, similar questions can be answered for other geometrical configuration of the roots of the polynomial or implicit conditions on the polynomial. Such conditions could be for example, what is the geometry of the control polygon when the polynomial has all its roots in a half-plane, or when the polynomial is univalent in a disk and so on. The most important fact is that every information on the geometry of the control polygon has an associated Bernstein type inequality and even an integral Bernstein type inequality. Due to space limitation, such a program has not been carried in this paper and it will be the theme of our forthcoming effort on the subject. We believe that studying the geometry of the control polygon of a complex Bézier curve will shed more light into different aspects of the geometry of polynomials, such as complex Rolle Theorem, root location algorithms and extremal problems.

Acknowledgement. This work was partially supported by the MEXT Global COE project. Osaka University, Japan.

References

1. Ait-Haddou, R., Nomura, T., Biard, L.: A refinement of the variation diminishing property of Bézier curves. Comput. Aided Geom. Design 27(2), 202–211 (2010)
2. Ait-Haddou, R., Herzog, W., Nomura, T.: Complex Bézier curves and the geometry of polygons. Comput. Aided Geom. Design 27(7), 525–537 (2010)
3. Aziz, A.: On the zeros of composite polynomials. Pacific J. Math. 103(1), 1–7 (1982)
4. Aziz, A.: On the location of the zeros of certain composite polynomials. Pacific J. Math. 118(1), 17–26 (1985)
5. de Bruijn, N.G.: Inequalities concerning polynomials in the complex domain Nederl. Akad. Wetensch. Proc. 50, 1265–1272 (1947)
6. Farin, G.: Curves and Surfaces for CAGD. A practical Guide, 5th edn. The Morgan Kaufmann Series in Computer Graphics Series (2002)
7. Goldman, R.N.: Dual polynomial bases. J. Approx. Theory 79, 311–346 (1994)
8. Govil, N.K., Mohapatra, R.N.: Markov and Bernstein type inequalities for polynomials. J. of lnequal. and Appl. 3, 349–387 (1999)
9. Jain, V.K.: Generalizations of parts of Grace's apolarity theorem involving circular regions (with a characteristic) and their applications. Complex Variables and Elliptic Equations 50(5), 357–366 (2005)
10. Lester, J.A., Triangles, I.: Shapes. Aequation Math. 52, 30–54 (1996)
11. Marden, M.: Geometry of Polynomials. Amer. Math. Soc., Providence (1966)
12. Prasolov, V.V.: Polynomials. Springer, Berlin (2004)
13. Ramshaw, L.: Blossoms are polar forms. Comput. Aided Geom. Design 6(4), 323–358 (1989)

The Shape of Conchoids to Plane Algebraic Curves

Juan Gerardo Alcázar*

Departamento de Matemáticas, Universidad de Alcalá,
E-28871-Madrid, Spain
juange.alcazar@uah.es

Abstract. The classical conchoid construction, well-known in Mathematics since its introduction more than 2200 years ago, has been recently revisited in the context of algebraic curves. Partially, the motivation for this has been the analogy, up to a certain extent, between the conchoid and a well-known transformation in CAGD, namely the offset transformation. In this paper, we contribute to this study by addressing properties on the shape of conchoids to plane algebraic curves. For this purpose we introduce the notions of *exterior* and *interior* conchoids, and we compare the shapes of these objects with that of the original curve.

1 Introduction

Given a curve $\mathcal{C}$, a fixed point F (the focus), and a distance d, the conchoid of $\mathcal{C}$ is the curve obtained from $\mathcal{C}$ by applying the following geometric transformation: for each point $P \in \mathcal{C}$, (i) trace the line connecting F, P; (ii) mark on this line the points P_1, P_2 lying at a distance d of P. Classical examples of this construction are the Conchoid of Nicomedes, where $\mathcal{C}$ is a line, and Pascal's Conchoids, where $\mathcal{C}$ is a circle. Recently, this construction has been revisited in several papers. More precisely, algebraic properties of conchoids of algebraic curves have been considered in [11], rationality questions have been discussed in [12], and algebraic properties of a more general class of transformations generalizing the conchoid construction have been addressed in [1].

The purpose of this paper is to contribute to the study of conchoids of plane algebraic curves by addressing topological questions. For this purpose, one takes advantage of the analogy between the conchoid construction considered here, and a well-known transformation arising in the context of Computer Aided Geometric Design, namely the *offset* transformation (see [7,9,10] for further reading on offsets and their properties). One may notice this analogy if one examines the equations of both transformations in vector form:

$$\bar{\boldsymbol{r}}_{\boldsymbol{c}} = \boldsymbol{r} \pm d\frac{\boldsymbol{r}}{\|\boldsymbol{r}\|},\ \ \bar{\boldsymbol{r}}_{\boldsymbol{o}} = \boldsymbol{r} \pm d\frac{\boldsymbol{N}}{\|\boldsymbol{N}\|}$$

* Author supported by the Spanish " Ministerio de Educación y Ciencia" under the Project MTM2008-04699-C03-01.

(here, $\boldsymbol{r}$ stands for the position vector of each point in the original curve, $\boldsymbol{N}$ is the normal vector, and $\boldsymbol{r}_c, \boldsymbol{r}_o$ stand for the corresponding points in the conchoid and the offset, respectively) Questions on the offset shape have been addressed in [2,3,4,5,7,8]. In these papers, one considers the offset to be the union of two subsets, called the *exterior* and the *interior* offsets, respectively, and applies different techniques to predict the topological behavior of the offset just from the original curve (i.e. without making use of the offset equation). In our case, we proceed in an analogous way. First, we define the exterior and interior conchoids; then we use the notion of *local shape* (introduced in [2,3,5] to study local properties of the offset, and used also in [4] to address global aspects of the offset shape) in order to study some local phenomena, and finally we address global properties. In this sense, we provide necessary and/or sufficient conditions for the exterior and/or the interior conchoids, or even the whole conchoid in certain cases, to be homeomorphic to the original curve.

The structure of this paper is the following. Preliminary questions (a brief review of the notion of local shape, and general results on the exterior and the interior conchoids) are provided in Section 2. Some local aspects on the conchoid shape are addressed in Section 3. Global questions are considered in Section 4. Finally, conclusions and comments on further work are presented in Section 5.

2 Preliminaries

2.1 Local Shape

Let $\mathcal{C}$ be an algebraic curve, and let $P = (x_0, y_0)$ be a real non-isolated point of $\mathcal{C}$. Then P is the *center* of at least one *real place* (see [13] for more information on places) of the curve, i.e. a local parametrization $\mathcal{P}(h) = (x(h), y(h))$ of $\mathcal{C}$ around P where $x(h), y(h)$ are real analytic functions, and $x(0) = x_0, y(0) = y_0$. Denoting by $I \subset \mathbb{R}$ the interval where $x(h), y(h)$ (that will be referred to as the *coordinates* of the place) are convergent, the set of real points of $\mathcal{C}$ described by $\mathcal{P}(h)$ for $h \in I$ is called a *real branch* of $\mathcal{C}$. Now in [3], places are used to formally describe the shape of a real branch by means of the notion of local shape. For the convenience of our readers, we briefly recall this notion here.

Given a real place $\mathcal{P}(h)$ centered at $P \in \mathcal{C}$, a $\mathcal{P}(h)$-*standard system* is a perpendicular system of coordinates where the origin is P, and where the x-axis is parallel to the tangent to $\mathcal{P}(h)$ at its center. In such a system, $\mathcal{P}(h)$ can be written (see [3]) in its so-called *standard form*, as

$$\mathcal{P}(h) = (h^p, \beta_q h^q + \gamma_r h^r + \cdots),$$

where $p, q \in \mathbb{N}$, and $1 \leq p < q$; the pair (p, q) is called the signature of the place (in fact, in [3] the place $\mathcal{P}(h)$ is written as $\mathcal{P}(h) = (\alpha_p h^p, \beta_q h^q + \gamma_r h^r + \cdots)$, but we can get rid of α_p by considering a simple change of parameter). Then $\mathcal{P}(h)$ can exhibit four different "shapes", shown in Figure 1, denoted as: local shape (I), or a thorn; (II), or an elbow; (III), or a beak; (IV), or a flex. Notice that since there are algorithms for computing places starting from the implicit equation of a curve and a point in it (see [13]), we can compute the local shape of a particular place by determining p, q and then checking whether p, q are even or odd, respectively.

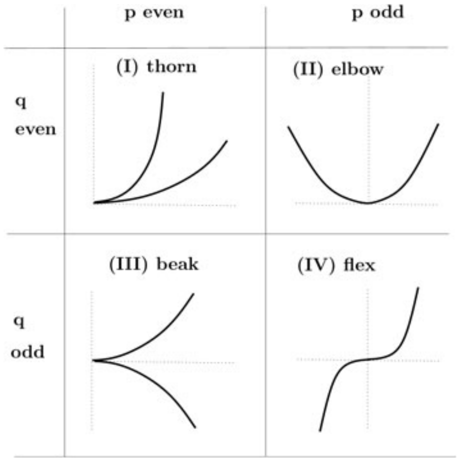


Fig. 1. Local Shapes

We will say that p is the *order of* $\mathcal{P}(h)$, i.e. the least power of h arising in the components $x(h), y(h)$ of $\mathcal{P}(h)$. Also, we will say that a place is *regular* if $p = 1$; otherwise we will say that it is *singular*. Notice that a regular point of $\mathcal{C}$ is always the center of a regular place. Singularities of $\mathcal{C}$ correspond to points which are either the center of several different places, or of just one, singular, place. In tthe first case we say that the curve has a *self-intersection* at the point: non-ordinary, if some places share a same tangent, ordinary if all the tangents are distinct. Finally, we will say that a place $\mathcal{P}(h)$ is cuspidal if p is even (in which case $\mathcal{P}(h)$ is either a thorn, or a beak; notice that in this case the center of the place is a (cusp-like) singularity of $\mathcal{C}$.

2.2 Exterior and Interior Conchoids

The conchoid to a curve $\mathcal{C}$ (the base curve) from focus F and distance d, is usually defined as the geometrical locus of all the points Q of the form

$$Q = P \pm d \cdot \frac{\boldsymbol{FP}}{\|\boldsymbol{FP}\|},$$

with $P \in \mathcal{C}$. We will refer to the geometric construction that lies behind this definition as the *original conchoid construction*. A more algebraic definition of conchoid is given in [11] for the case when $\mathcal{C}$ is an algebraic curve: in that situation, the conchoid of $\mathcal{C}$ from focus $F = (a_1, a_2) \in \mathbb{C}^2$ and distance $d \in \mathbb{C}$, represented as $\mathfrak{C}_{d,F}(\mathcal{C})$, is defined as the Zariski closure of the points (i.e. the smallest algebraic curve containing the points) computed by applying the original conchoid construction to $\mathcal{C}\backslash\mathcal{C}_0$, where

$$\mathcal{C}_0 = \{P \in \mathcal{C} / \|\boldsymbol{FP}\| = 0\}$$

With this definition, the implicit equation of $\mathfrak{C}_{d,F}(\mathcal{C})$ can be computed by means of elimination methods like the Gröbner basis, or resultants, see [11,1]. However, as it also happens in the case of offsets, this equation tends to be bigger than that of the base curve. Hence, it is preferable to determine the properties of the conchoid, without manipulating or even computing its equation.

In the rest of the paper we will use the above notion for $\mathfrak{C}_{d,F}(\mathcal{C})$. Furthermore, we will assume that F and d are fixed, and hence we will represent the conchoid as $\mathfrak{C}(\mathcal{C})$, without explicitly spelling the focus or the distance. We will also assume that the curve $\mathcal{C}$ we work with is a real curve; in fact, we will be interested in analyzing the real part of $\mathfrak{C}(\mathcal{C})$. Finally, we will also assume that $d \in \mathbb{R}$, $d > 0$, that $F \in \mathbb{R}^2$ and in fact, without loss of generality, that $F = (0, 0)$.

Now let us introduce two mappings: for $\sigma \in \{-1, +1\}$ let

$$\begin{aligned} &\mathcal{C}\backslash\mathcal{C}_0 \overset{\psi_{\sigma,d}}{\longmapsto} \mathfrak{C}(\mathcal{C}) \\ &(x, y) \longmapsto (X, Y) = \psi_{\sigma,d}(x, y) = \left(x + d\sigma \frac{x}{\sqrt{x^2+y^2}}, y + d\sigma \frac{y}{\sqrt{x^2+y^2}} \right) \end{aligned}$$

We will write $\psi_{+1,d} = \psi_{+d}$ and $\psi_{-1,d} = \psi_{-d}$, respectively. Notice that since we are assuming that $F = (0, 0)$, then $\sqrt{x^2+y^2}$ represents the distance between the focus and any point of $\mathcal{C}\backslash\mathcal{C}_0$. Then, ψ_{+d}, ψ_{-d} are defined for all the real points of $\mathcal{C}$, except for F in the case when $F \in \mathcal{C}$. These mappings allow us to introduce the following definition, that will be essential for our purposes.

Definition 1. *The* Exterior Conchoid of $\mathcal{C}$*,* $\mathfrak{C}_{ext}(\mathcal{C})$ *(resp. the* Interior Conchoid of $\mathcal{C}$*,* $\mathfrak{C}_{int}(\mathcal{C})$*) is the set consisting of all the points* $Q \in \mathfrak{C}(\mathcal{C})$ *of the form* $Q = \psi_{+d}(P)$ *(resp.* $Q = \psi_{-d}(P)$*) for some point* $P \in \mathcal{C}$*.*

According to the definition of conchoid given in [11], $\mathfrak{C}(\mathcal{C})$ is $\mathfrak{C}_{int}(\mathcal{C}) \cup \mathfrak{C}_{ext}(\mathcal{C})$ together with the points added when taking the Zariski closure of this last set, which, by the Closure Theorem (Theorem 3 in [6], see p. 125), consists just of finitely many points. The geometric construction lying behind the above definition is illustrated in Figure 2. Here, we have denoted $P_{+d} = \psi_{+d}(P)$, $P_{-d} = \psi_{-d}(P)$. At the left of Figure 2, we have the case when the distance $\text{dis}(P, F)$ between the focus F and the point P is greater than d; at the right, we have the case when $\text{dis}(P, F) < d$.

In Figure 3 we can see $\mathfrak{C}_{ext}(\mathcal{C})$ and $\mathfrak{C}_{int}(\mathcal{C})$ for two algebraic curves, namely the cardioid (left) and the epitrochoid (right). In both cases, $\mathfrak{C}_{ext}(\mathcal{C})$ is plotted in thick solid line, and $\mathfrak{C}_{int}(\mathcal{C})$ in thick dotted line. We will follow the same criterion (i.e. solid line for $\mathfrak{C}_{ext}(\mathcal{C})$, dotted line for $\mathfrak{C}_{int}(\mathcal{C})$) in all the pictures of the paper. In the case of the cardioid, we have fixed $F = (0. - 7.5)$, and $d = 4.7$; for the epitrochoid, $F = (0, 1.6)$ and $d = 0.3$.

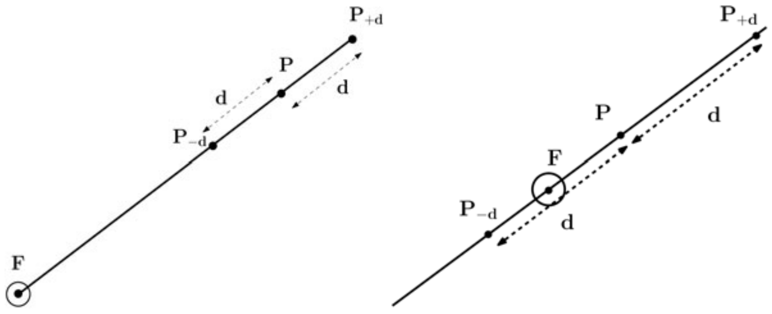


Fig. 2. Constructions for $\mathfrak{C}_{ext}(\mathcal{C})$ and $\mathfrak{C}_{int}(\mathcal{C})$s

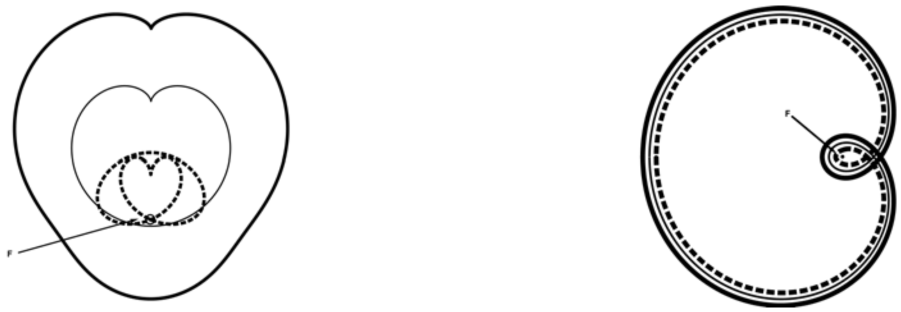


Fig. 3. Exterior and Interior Conchoids of the Cardioid (left) and the Epitrochoid (right)

Although $\mathcal{C}$, $\mathfrak{C}(\mathcal{C})$, etc. are in principle objects of $\mathbb{C}^2$, we will be interested in studying their real parts. For this purpose, we need to recall some more results of [11]. In that paper it is proven that $\mathfrak{C}(\mathcal{C})$ has at most two components, and that whenever $\mathcal{C}$ is not a circle of center F and radius d, these components have dimension 1 (i.e. they do not degenerate into 0-dimensional subsets). Also in [11] a component $\mathcal{M}$ of $\mathfrak{C}(\mathcal{C})$ is said to be special if the points of $\mathcal{M}$ are generated by more than one point of the original curve. It is also shown that given $\mathcal{C}$ there are only finitely many distances such that $\mathfrak{C}(\mathcal{C})$ has some special component. So, in the sequel we will also assume that $\mathcal{C}$ is different from a line, and that $\mathfrak{C}(\mathcal{C})$ has no degenerated or special components. Under this last assumption one may see that there are just finitely real many points of $\mathfrak{C}(\mathcal{C})$ which are generated, via ψ_{-d}, ψ_{+d}, by complex points of $\mathcal{C}$. So, one can prove the following theorem, which makes clear the relationship between the real parts of the objects of our interest. Here $\mathcal{F}$ denotes the set of real points generated in $\mathfrak{C}(\mathcal{C})$ by F; so, $\mathcal{F} = \emptyset$ when $F \notin \mathcal{C}$.

Theorem 1. *If $\mathfrak{C}(\mathcal{C})$ has no special component, then the points of $\mathfrak{C}(\mathcal{C}) \cap \mathbb{R}^2$ are generated by the points of $\mathcal{C} \cap \mathbb{R}^2$. Furthermore, discarding isolated singularities,*

$$\mathfrak{C}(\mathcal{C}) \cap \mathbb{R}^2 - \mathcal{F} = \left(\mathfrak{C}_{int}(\mathcal{C}) \cap \mathbb{R}^2\right) \cup \left(\mathfrak{C}_{ext}(\mathcal{C}) \cap \mathbb{R}^2\right).$$

The above theorem provides the following corollary.

Corollary 1. *The only points of $\mathfrak{C}(\mathcal{C}) \cap \mathbb{R}^2$ which are not of the form $\psi_{+d}(P)$ or $\psi_{-d}(P)$ for some real point $P \in \mathcal{C}$, are the points generated by the focus, in the case when it is on $\mathcal{C}$.*

For simplicity, in the sequel whenever we write $\mathcal{C}$, $\mathfrak{C}(\mathcal{C})$, we will mean the real parts of these objects, discarding isolated singularities. Furthemore, whenever we write $\mathfrak{C}_{int}(\mathcal{C})$, $\mathfrak{C}_{ext}(\mathcal{C})$, we will mean the topological closures (in the usual topology of $\mathbb{R}^2$) of the real parts of these objects, discarding isolated singularities. The reason for taking the closures is to include the points of $\mathcal{F}$ and therefore to avoid cumbersome statements.

3 Local Analysis around the Focus

In order to see how the conchoid construction affects the shape of $\mathcal{C}$ (which is basically the purpose of next section), one has to separately study the cases when $F \notin \mathcal{C}$ and $F \in \mathcal{C}$. In this last case, it is necessary to know the effect of ψ_{+d}, ψ_{-d} on the vicinity of F. This can be done by making use of the notion of *local shape*, recalled in Subsection 2.1. So, assume that $F \in \mathcal{C}$, and let $\mathcal{P}(h)$ be a real place centered at F, given in standard form. Since we are assuming that $F = (0,0)$, we have that $\mathcal{P}(h) = (h^p, \beta_q h^q + \cdots)$. So, by applying ψ_{+d}, ψ_{-d} to $\mathcal{P}(h)$ and making computations with power series (as in [3]), we can determine the conchoid places that $\mathcal{P}(h)$ generates. First, we get that:

$$\frac{1}{\sqrt{x(h)^2 + y(h)^2}} = \frac{1}{|h^p|} \cdot \left(1 - \frac{\beta_q^2}{2} h^{2(q-p)} + \cdots\right)$$

Substituting the above expression into $\psi_{+d}(\mathcal{P}(h))$ and $\psi_{-d}(\mathcal{P}(h))$ we find the step function $\dfrac{h^p}{|h|^p}$. Then we get two real places of $\mathfrak{C}(\mathcal{C})$, that we denote as $\mathcal{P}_{+d}(h)$ and $\mathcal{P}_{-d}(h)$, that can be written (in a compact form) as

$$\mathcal{P}_{\pm d}(h) = \left(\pm d + h^p \mp \frac{d\beta_q^2}{2} h^{2(q-p)} + \cdots, \mp d\beta_q h^{q-p} \pm + \cdots\right) \tag{1}$$

Because of the behavior of $\dfrac{h^p}{|h|^p}$ when $h \to 0$, the equalities

$$\mathcal{P}_{+d}(h) = \psi_{+d}(\mathcal{P}(h)),\ \mathcal{P}_{-d}(h) = \psi_{-d}(\mathcal{P}(h))$$

hold if and only if p is even, i.e. iff $\mathcal{P}(h)$ is cuspidal. Otherwise we still get two places, but none of them is fully contained either in $\mathfrak{C}_{ext}(\mathcal{C})$ or in $\mathfrak{C}_{int}(\mathcal{C})$; in fact, in this case $\psi_{+d}(\mathcal{P}(h))$ and $\psi_{-d}(\mathcal{P}(h))$ "jump" from $\mathfrak{C}_{ext}(\mathcal{C})$ to $\mathfrak{C}_{int}(\mathcal{C})$ or conversely, as the focus is crossed. We summarize this in the following result.

Theorem 2. *Assume that $F \in \mathcal{C}$, and let $\mathcal{P}(h)$ be a real place of $\mathcal{C}$ centered at P. Also, let $\mathcal{P}_{\pm d}(h)$ denote the places generated by $\mathcal{P}(h)$ in $\mathfrak{C}(\mathcal{C})$. Then $\mathcal{P}_{+d}(h)$ (resp. $\mathcal{P}_{-d}(h)$) is fully contained in $\mathfrak{C}_{ext}(\mathcal{C})$ (resp. $\mathfrak{C}_{int}(\mathcal{C})$) if and only if $\mathcal{P}(h)$ is cuspidal.*

The situation in Theorem 2 is illustrated in Figure 4. Here we can see the conchoid of a circle (i.e. a *Pascal's Snail*) computed in the case when F is a point on the circle, and the distance d is bigger than the radius. The place $\mathcal{P}(h)$ gives rise to two conchoid places, but none of them lies completely either in $\mathfrak{C}_{int}(\mathcal{C})$ (in dots) or $\mathfrak{C}_{ext}(\mathcal{C})$ (in thick solid line). In fact, each of these places lie half in $\mathfrak{C}_{int}(\mathcal{C})$, and the other half in $\mathfrak{C}_{ext}(\mathcal{C})$.

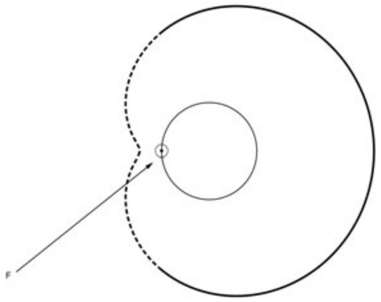


Fig. 4. Places generated by a circle place centered at the focus

On the other hand, from the expression (1) we can see that F generates the points $(d, 0)$ and $(-d, 0)$, expressed in a $\mathcal{P}(h)$-standard system (in fact these are the centers of the places $\mathcal{P}_{\pm d}(h)$). Thus if F is the center of several places of $\mathcal{C}$ which do not share a same tangent, we have different standard systems for different tangents, and hence F generates different points for different tangent directions. Otherwise if F is the center of several places of $\mathcal{C}$ sharing a same tangent (in which case F is a non-ordinary self-intersection of $\mathcal{C}$), then only two points in $\mathfrak{C}(\mathcal{C})$ are generated. In this situation, by Theorem 2, F gives rise to one self-intersection of $\mathfrak{C}_{ext}(\mathcal{C})$ and another self-intersection of $\mathfrak{C}_{int}(\mathcal{C})$, both with the same multiplicity than F, iff all these places are cuspidal. We summarize these reasonings in the following result.

Theorem 3. *Let $F \in \mathcal{C}$ be a self-intersection of $\mathcal{C}$ with multiplicity m (i.e. with m real branches of $\mathcal{C}$ intersecting there). Then the following statements are true:*

(i) F is never invariant under the conchoid transformation.
(ii) F generates a self-intersection of $\mathfrak{C}_{ext}(\mathcal{C})$ and a self-intersection of $\mathfrak{C}_{int}(\mathcal{C})$ both with multiplicity m, if and only if all the real places of $\mathcal{C}$ centered at F are cuspidal, and share the same tangent.

Remark 1. When $F \in \mathcal{C}$ and is the center of one real cuspidal place $\mathcal{P}(h)$, the functions ψ_{-d}, ψ_{+d} introduced in Subsection 2.2, that are defined over $\mathcal{C} - \{F\}$, can be extended to functions $\tilde{\psi}_{-d}, \tilde{\psi}_{+d}$, well-defined and continuous over $\mathcal{C}$. Indeed, in order to do this it suffices to define $\tilde{\psi}_{+d}(F)$ (resp. $\tilde{\psi}_{-d}(F)$) as the center of the place $\mathcal{P}_{+d}(h)$ (resp. $\mathcal{P}_{-d}(h)$). This is also true when $F \in \mathcal{C}$ is the center of several cuspidal places of $\mathcal{C}$ sharing a same tangent line.

4 Global Questions

In this section we consider global aspects on the topology of conchoids. Basically, the problem that we address is to find conditions so that the topology of $\mathcal{C}$ is kept invariant when computing $\mathfrak{C}_{ext}(\mathcal{C})$ or $\mathfrak{C}_{int}(\mathcal{C})$. This idea is made precise in the following definition.

Definition 2. *We say that $\mathfrak{C}_{ext}(\mathcal{C})$ (resp. $\mathfrak{C}_{int}(\mathcal{C})$ or $\mathfrak{C}(\mathcal{C})$) has a* good global behavior*, if it is homeomorphic to $\mathcal{C}$ (i.e. if their topologies coincide).*

The functions ψ_{+d}, ψ_{-d} can be used for studying this. Indeed, whenever ψ_{+d} (resp. ψ_{-d}) is a homeomorphism (i.e. continuous and with continuous inverse), or can be extended to a homeomorphism, of $\mathcal{C}$ onto $\mathfrak{C}_{ext}(\mathcal{C})$ (resp. $\mathfrak{C}_{int}(\mathcal{C})$), we can ensure that $\mathfrak{C}_{ext}(\mathcal{C})$ (resp. $\mathfrak{C}_{int}(\mathcal{C})$) has a good global behavior. Thus, our strategy will be: (i) address conditions for ψ_{+d} (resp. ψ_{-d}) to be a homeomorphism; (ii) study the cases when these conditions are also necessary.

We start studying when ψ_{+d}, ψ_{-d} are invertible. This is done in the following lemma, which can be proven by making elementary computations with ψ_{+d}, ψ_{-d}. The reader may also intuitively check the statement in the lemma by inspecting Figure 2 in Subsection 2.2.

Lemma 1. *Let $Q \in \mathfrak{C}(\mathcal{C})$, $Q \neq F$. Then the following statements are true:*

(i) If $Q \in \mathfrak{C}_{ext}(\mathcal{C})$ and there exists $P \in \mathcal{C}$ such that $Q = P_{+d}$, then $\psi_{+d}^{-1}(Q) = P = \psi_{-d}(Q)$.
(ii) If $Q \in \mathfrak{C}_{int}(\mathcal{C})$ and there exists $P \in \mathcal{C}$ such that $Q = P_{-d}$, then

$$\psi_{-d}^{-1}(Q) = P = \begin{cases} \psi_{+d}(Q) \text{ if } dis(\psi_{+d}(Q, F) \geq d \\ \psi_{-d}(Q) \text{ if } dis(\psi_{-d}(Q, F) < d \end{cases}$$

From the expressions of ψ_{+d} and ψ_{-d} (see Subsection 2.2) one may see that they are continuous at any point different from F. In order to ensure that they are homeomorphisms we have to check the continuity of their inverses as well. Since from the above lemma these inverses involve also ψ_{+d} and ψ_{-d}, it is important to know whether F belongs to $\mathfrak{C}(\mathcal{C})$, or not. For this purpose, we consider the following result.

Lemma 2. *The focus F never belongs to $\mathfrak{C}_{ext}(\mathcal{C})$. Moreover, it belongs to $\mathfrak{C}_{int}(\mathcal{C})$ if and only if there exists $P \in \mathcal{C}$ such that $dis(P, F) = d$.*

Proof. From the original conchoid construction it holds that if $P \in \mathcal{C}$, $P \neq F$, generates $P' \in \mathfrak{C}_{ext}(\mathcal{C})$, then $\text{dis}(P', F) > \text{dis}(P, F)$, and therefore $P' \neq F$; furthermore, if $F \in \mathcal{C}$, from the statement (ii) in Theorem 3 we deduce that it does not generate F, either. Hence, we conclude that $F \notin \mathfrak{C}_{ext}(\mathcal{C})$. On the other hand, if there exists some non-isolated real point $P \in \mathcal{C}$ such that $\text{dis}(P, F) = d$, then $\psi_{-d}(P) = F$ and therefore $F \in \mathfrak{C}_{int}(\mathcal{C})$. Conversely, let $F \in \mathfrak{C}_{int}(\mathcal{C})$. Since F is not invariant when computing the conchoid, from Corollary 1 there exists $P \in \mathcal{C}$ such that $\psi_{-d}(P) = F$, and therefore $\text{dis}(P, F) = d$. □

Now let us provide conditions for ψ_{+d} to be a homeomorphism. This is done in the following proposition.

Proposition 1. *Assume that one of the following conditions happen: (1) $F \notin \mathcal{C}$; (2) $F \in \mathcal{C}$ and is the center of just one real, cuspidal, place of $\mathcal{C}$; (3) $F \in \mathcal{C}$ is the center of several cuspidal places all of them sharing a same tangent. Then, either ψ_{+d} is a homeomorphism over $\mathcal{C}$, or it can be extended to a homeomorphism over $\mathcal{C}$.*

Proof. Whenever $F \notin \mathcal{C}$, ψ_{+d} is continuous over $\mathcal{C}$. Furthermore, if $F \in \mathcal{C}$ is the center of just one cuspidal real place, or the center of several cuspidal places sharing a same tangent, then from Remark 1 one may see that ψ_{+d} can be extended to a continuous function over $\mathcal{C}$. On the other hand, since by Lemma 1 it holds that $\psi_{+d}^{-1} = \psi_{-d}$, and by Lemma 2 we have that $F \notin \mathfrak{C}_{ext}(\mathcal{C})$, then ψ_{+d}^{-1} is always continuous over $\mathfrak{C}_{ext}(\mathcal{C})$. □

We are going to prove that the conditions in Proposition 1 are also necessary for the topologies of $\mathcal{C}$ and $\mathfrak{C}_{ext}(\mathcal{C})$ to coincide. For this purpose, first we need the following lemma. This result states that $\mathfrak{C}_{ext}(\mathcal{C})$ has no other self-intersections apart from those corresponding to the self-intersections of $\mathcal{C}$.

Lemma 3. *Every self-intersection of $\mathcal{C}$, different from the focus, gives rise to a self-intersection of $\mathfrak{C}_{ext}(\mathcal{C})$. Conversely, every self-intersection of $\mathfrak{C}_{ext}(\mathcal{C})$ comes from a self-intersection of $\mathcal{C}$.*

Proof. The implication $(\Rightarrow)$ follows from the conchoid construction. So, let us see $(\Leftarrow)$. For this purpose, by Lemma 1 it holds that $\psi_{+d}^{-1} = \psi_{-d}$. Now, given $Q \in \mathfrak{C}_{ext}(\mathcal{C})$, by Lemma 2 it follows that $Q \neq F$, and therefore $\psi_{+d}^{-1}(Q)$ is well defined. Finally, by the conchoid construction if Q is a self-intersection, then $\psi_{+d}^{-1}(Q) \in \mathcal{C}$ is also a self-intersection of $\mathcal{C}$. Hence, the statement holds. □

Then we are finally ready for the following theorem on the good global behavior of $\mathfrak{C}_{ext}(\mathcal{C})$.

Theorem 4. *$\mathfrak{C}_{ext}(\mathcal{C})$ has a good global behavior if and only if one of the following situations happen: (1) $F \notin \mathcal{C}$; (2) $F \in \mathcal{C}$ and is the center of just one real, cuspidal, place of $\mathcal{C}$; (3) $F \in \mathcal{C}$ is the center of several cuspidal places all of them sharing a same tangent.*

Proof. The statement ($\Leftarrow$) is Proposition 1. So, we just have to prove ($\Rightarrow$). Indeed, if $\mathfrak{C}_{ext}(\mathcal{C})$ is homeomorphic to $\mathcal{C}$ then either $F \notin \mathcal{C}$ (and (1) happens), or $F \in \mathcal{C}$. Let us see that in this last case either (2) or (3) must occur. Assume first that F is the center of just one real place $\mathcal{P}(h)$, and let $\mathcal{Q}(h)$ be the place of $\mathfrak{C}(\mathcal{C})$ generated by $\mathcal{P}(h)$; also, let Q be the center of $\mathcal{Q}(h)$. Since Q is generated by F, and F is not a self-intersection, then by Lemma 3 Q is not a self-intersection, either. Now if $\mathcal{P}(h)$ is not cuspidal, then by Theorem 2 just half of $\mathcal{Q}(h)$ is contained in $\mathfrak{C}_{ext}(\mathcal{C})$. Furthermore, since Q is not a self-intersection of $\mathfrak{C}_{ext}(\mathcal{C})$ we get that $\mathfrak{C}_{ext}(\mathcal{C})$ has a branch at Q which is not continued. But then it cannot be homeomorphic to $\mathcal{C}$ (which is the real zero-set of an algebraic curve, discarding isolated singularities). Hence, (2) happens. Finally, assume that F is the center of several real places. If these places are not cuspidal, and sharing a same tangent, then by Theorem 3 we cannot have a self-intersection of $\mathfrak{C}_{ext}(\mathcal{C})$ with the same multiplicity as F (in $\mathcal{C}$). Hence, if (3) does not occur, then $\mathfrak{C}_{ext}(\mathcal{C})$ cannot be homeomorphic to $\mathcal{C}$. So, (3) must happen. $\square$

Now let us consider $\mathfrak{C}_{int}(\mathcal{C})$. For this purpose, as in the above case we start giving conditions for ψ_{-d} to be homeomorphism. In this sense, the following lemma, that can be easily proven, is needed.

Lemma 4. *The following statements are true:*

(i) Every self-intersection of $\mathcal{C}$, different from the focus, gives rise to a self-intersection of $\mathfrak{C}_{int}(\mathcal{C})$.

(ii) Conversely, if Q is a self-intersection of $\mathfrak{C}_{int}(\mathcal{C})$, then one of the following statements hold: (a) Q is generated by a self-intersection of $\mathcal{C}$; (b) Q is the focus; moreover, this happens iff there exist at least two different points $P, P' \in \mathcal{C}$ such that $dis(P,F) = dis(P,F') = d$; (c) Q is generated by two different points $P, P' \in \mathcal{C}$, placed at different sides of the focus (i.e. $\boldsymbol{FP} \cdot \boldsymbol{FP'} < 0$), such that $dis(P,Q) = dis(P',Q)$ and $dis(P,P') = 2d$.

We refer to the points fulfilling the condition (c) in the statement (ii) of the above lemma, as *special self-intersections.* This kind of self-intersections is illustrated in Figure 5.

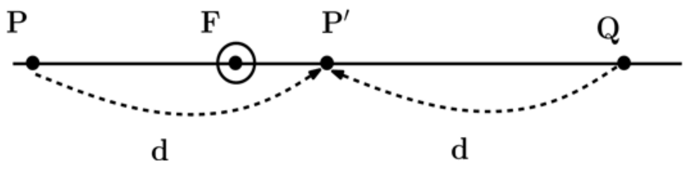


Fig. 5. Self-intersection of the Interior Conchoid

Special self-intersections are important because if Q is such a point, then in a vicinity of Q, the mapping ψ_{-d}^{-1} is not continuous: indeed, from Lemma 1 in that case ψ_{-d}^{-1} is a piecewise function and the point where the expression of ψ_{-d}^{-1}

changes (in fact, in a non-continuous way) is Q. Furthermore, we will speak about *additional self-intersections* to mean self-intersections of $\mathfrak{C}_{int}(\mathcal{C})$ which are not generated by self-intersections of the base curve. Now we also need the following result, that has to do with the behavior of ψ_{-d}^{-1} around the focus in one particular case.

Lemma 5. *If there is just one non-isolated point $P \in \mathcal{C}$ fulfilling that $dis(P,F) = d$ (in which case the line connecting P, F is normal to $\mathcal{C}$ at P), then $\psi_{-d}^{-1}|_{\mathfrak{C}_{int}(\mathcal{C})}$ is continuous in the vicinity of the focus.*

Proof. Since P is not isolated then it is the center of a real place $\mathcal{P}(h)$ of $\mathcal{C}$. Since $dis(P,F) = d > 0$ then $P \neq F$, and therefore $\psi_{-d}(\mathcal{P}(h)) = \mathcal{Q}(h)$ is a place contained in $\mathfrak{C}_{int}(\mathcal{C})$, centered at F. Furthermore, by Lemma 2 then P is the only point of $\mathcal{C}$ transforming onto the focus. Thus, there exists a neighborhood $E \subset \mathbb{R}^2$ of P such that $\mathfrak{C}_{int}(\mathcal{C}) \cap E = \mathcal{Q}(h)$. Since $\psi_{-d}^{-1}(\mathcal{Q}(h)) = \mathcal{P}(h)$, we deduce that $\psi_{-d}^{-1}|_{\mathfrak{C}_{int}(\mathcal{C}))\cap E}$ is continuous. □

Hence, we can provide conditions for ψ_{-d} to be a homeomorphism. This is done in the following proposition.

Proposition 2. *Assume that there are no additional self-intersections. Assume also that one of the following conditions happen: (1) $F \notin \mathcal{C}$; (2) $F \in \mathcal{C}$ and is the center of just one real, cuspidal, place of $\mathcal{C}$; (3) $F \in \mathcal{C}$ is the center of several cuspidal places all of them sharing a same tangent. Then, either ψ_{-d} is a homeomorphism over $\mathcal{C}$, or it can be extended to a homeomorphism over $\mathcal{C}$.*

Proof. If (1) holds, then ψ_{-d} is continuous. Furthermore, since by hypothesis there are no additional self-intersections, then either $F \notin \mathfrak{C}_{int}(\mathcal{C})$, or $F \in \mathfrak{C}_{int}(\mathcal{C})$ but it is the center of just one real place. In the first case ψ_{-d}^{-1} is continuous around the focus; the same happens in the second case, because of Lemma 5. Since by hypothesis there are no special self-intersections, then ψ_{-d}^{-1} is continuous over $\mathfrak{C}_{int}(\mathcal{C})$, and therefore ψ_{-d} is a homeomorphism. (2) can be proven in a similar way, taking also into account Theorem 2. (3) can also be proven similarly, taking also into account Theorem 3. □

Now let us prove that the conditions in Proposition 2 are also necessary, in some cases, for $\mathfrak{C}_{int}(\mathcal{C})$ to have a good global behavior. First, we have the following result on $\mathfrak{C}_{int}(\mathcal{C})$, for the case when $F \notin \mathcal{C}$.

Theorem 5. *Assume that $F \notin \mathcal{C}$. Then, $\mathfrak{C}_{int}(\mathcal{C})$ has a good global behavior if and only no additional self-intersections occur.*

Proof. ($\Leftarrow$) follows from Proposition 2. ($\Rightarrow$) can be proven as in Theorem 4. □

In the case when $F \in \mathcal{C}$, but F is not a self-intersection of $\mathcal{C}$, the following theorem, that can be proven as Theorem 5, holds.

Theorem 6. *Let $F \in \mathcal{C}$, and assume that F is the center of just one real place $\mathcal{P}(h)$ of $\mathcal{C}$. Then, $\mathfrak{C}_{int}(\mathcal{C})$ has a good global behavior if and only if $\mathcal{P}(h)$ is cuspidal, and no additional self-intersections occur.*

Finally, in the case when $F \in \mathcal{C}$ is a self-intersection of $\mathcal{C}$ it is not true in general that the conditions in Proposition 2 are necessary. So, we have the following result.

Theorem 7. *Assume that $F \in \mathcal{C}$, and it is a self-intersection of $\mathcal{C}$. Then, the following statements are true.*

(i) If $\mathfrak{C}_{int}(\mathcal{C})$ has a good global behavior, then every place of $\mathcal{C}$ centered at F must be cuspidal.
(ii) If F is the center of several cuspidal places of $\mathcal{C}$, all of them sharing a same tangent, and no additional self-intersections occur, then $\mathfrak{C}_{int}(\mathcal{C})$ has a good global behavior.

Proof. The statement (i) follows from Theorem 3. The statement (ii) is Proposition 2. □

Let us see that the converse statements of Theorem 7 do not hold. Indeed, in Figure 6, right, one may see the picture of the algebraic curve defined by $f(x, y) := (16y^2 - 4x^3 + x^4)(16x^2 - 4y^3 + y^4) = 0$, together with its interior conchoid, for $F = (0, 0)$ and $d = 3$. Notice that the origin is the center of two cuspidal places, but however $\mathfrak{C}_{int}(\mathcal{C})$ and $\mathcal{C}$ are not homeomorphic; so, the converse of the statement (i) of the theorem does not hold. Furthermore, also in Figure 6, at the left, we have plotted the curve together with its interior conchoid, for $F = (0, 0)$ and $d = 4$. In this case, both objects are homeomorphic. However, the places of $f(x, y)$ centered at the origin do not share a same tangent, and therefore the converse of the statement (ii) of Theorem 7 does not hold. What is happening here is that although the self-intersection of the base curve at the focus is "lost" when computing the interior conchoid, another self-intersection, in fact an *additional* self-intersection, is gained; so, in the end the topology is kept invariant.

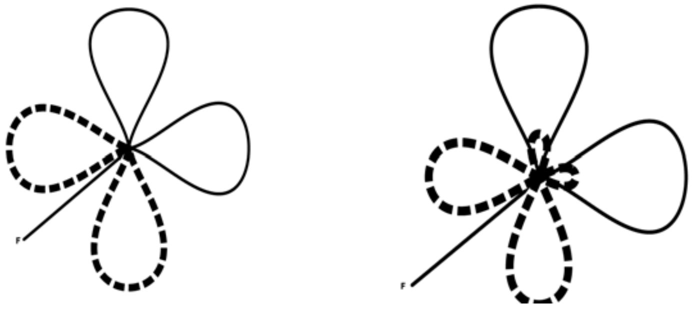


Fig. 6. Counterexamples to the converse statements of Theorem 7

We end this section by analyzing the special case when $\mathcal{C}$ is a closed curve homeomorphic to a circle, and the focus is a point of $\mathcal{C}$. If F is the center of a cuspidal place, then the analysis follows from the above results. So, in the sequel we address the case when F is the center of a non-cuspidal place of $\mathcal{C}$. In that situation, neither $\mathfrak{C}_{ext}(\mathcal{C})$ nor $\mathfrak{C}_{int}(\mathcal{C})$ have a good global behavior. However, as it happens in the case of Figure 4, $\mathfrak{C}(\mathcal{C})$ *can* have a good global behavior. We begin with the following lemma, concerning the intersections between the exterior and the interior conchoid.

Lemma 6. *Let $P, P' \in \mathcal{C}$, $P \neq F$, $P' \neq F$. Then P, P' generate a common point of $\mathfrak{C}_{ext}(\mathcal{C})$ and $\mathfrak{C}_{int}(\mathcal{C})$ if and only if P, P', F are aligned, $dis(P, P') = 2d$ and F does not lie in between P, P' (i.e. $\boldsymbol{FP} \cdot \boldsymbol{FP'} > 0$).*

Moreover we also need the following result.

Lemma 7. *Let $\mathcal{C}$ be a closed curve homeomorphic to a circle and let $F \in \mathcal{C}$ be the center of just one real, non-cuspidal, place. Then, $\mathfrak{C}(\mathcal{C})$ is a closed and connected curve.*

Proof. Since $\mathcal{C}$ is homeomorphic to a circle and $F \in \mathcal{C}$, then $\mathcal{C} - \{F\}$ is connected. Thus, since ψ_{+d} is continuous over $\mathcal{C} - \{F\}$, we have that $\psi_{+d}(\mathcal{C} - \{F\})$ is also a connected set. In fact, since F is the center of just one real, non-cuspidal, place $\mathcal{P}(h)$, by Theorem 3 it gives rise to two different points, A, B, which are joined by the closure $\overline{\psi_{+d}(\mathcal{C} - \{F\})}$. Similarly, $\psi_{-d}(\mathcal{C} - \{F\})$ is a connected set, and $\overline{\psi_{-d}(\mathcal{C} - \{F\})}$ also connects A, B. Finally, since A, B both belong to $\overline{\psi_{+d}(\mathcal{C} - \{F\})}$ and $\overline{\psi_{-d}(\mathcal{C} - \{F\})}$, and $\mathfrak{C}(\mathcal{C})$ is the union of these two curves, we deduce that $\mathfrak{C}(\mathcal{C})$ is a closed curve. Moreover, it is connected because it is the union of two connected subsets with a common point (in fact, with two points in common, A and B) $\square$

So, we finally deduce the following theorem, which follows from Lemma 6 and Lemma 7.

Theorem 8. *Assume that $\mathcal{C}$ is a closed curve, homeomorphic to a circle, and such that $F \in \mathcal{C}$ is the center of just one real place $\mathcal{P}(h)$ with signature (p, q), where p is odd. Then, $\mathfrak{C}(\mathcal{C})$ has a good global behavior if and only if the following two conditions hold: (1) $\mathfrak{C}_{ext}(\mathcal{C})$ and $\mathfrak{C}_{int}(\mathcal{C})$ do not have common points; (2) $\mathfrak{C}_{int}(\mathcal{C})$ has no additional self-intersections.*

Proof. Since $\mathcal{C}$ has no self-intersections, then (1) and (2) are necessary conditions. So, let us see that they are also sufficient. By Lemma 7 we have that $\mathfrak{C}(\mathcal{C})$ is closed and connected. Furthermore, since (1) and (2) hold then it has no self-intersections, and thus we conclude that it is homeomorphic to a circle. $\square$

5 Conclusions and Further Work

We have presented results that allow to predict, under certain hypotheses, some features of the topology of $\mathfrak{C}(\mathcal{C})$ from the topology of $\mathcal{C}$. In the case when $F \notin \mathcal{C}$, the study is relatively easy. However, when $F \in \mathcal{C}$ the problem is more difficult, and requires to address the local behavior of the conchoid transformation around

F. This can be done by means of the notion of "local shape", already used in the offset case. Using this tool, we provide necessary and sufficient conditions for a good global behavior of: (i) $\mathfrak{C}_{ext}(\mathcal{C})$; (ii) $\mathfrak{C}_{int}(\mathcal{C})$, in certain cases; (iii) $\mathfrak{C}(\mathcal{C})$, when $\mathcal{C}$ is homeomorphic to a circle and $F \in \mathcal{C}$. We also provide sufficient conditions for $\mathfrak{C}_{int}(\mathcal{C})$ to have a good global behavior under more general hypotheses. The natural continuation of this paper is the study of conchoids of algebraic surfaces; however, in order to do that the necessary algebraic background on such surfaces has still to be developed.

Acknowledgements. The author wants to thank Fernando San Segundo for his help with the package Geogebra (`http://www.geogebra.org`), which has been a very useful tool to explore many of the questions addressed in the paper.

References

1. Albano, A., Roggero, M.: Conchoidal transform of two plane curves. Applicable Algebra in Engineering, Communication and Computing 21, 309–328 (2010)
2. Alcazar, J.G., Sendra, J.R.: Local Shape of Offsets to Rational Algebraic Curves, Tech. Report SFB 2006-22, RICAM, Austria (2006)
3. Alcazar, J.G., Sendra, J.R.: Local Shape of Offsets to Algebraic Curves. Journal of Symbolic Computation 42, 338–351 (2007)
4. Alcazar, J.G.: Good Global Behavior of Offsets to Plane Algebraic Curves. Journal of Symbolic Computation 43, 659–680 (2008)
5. Alcazar, J.G.: Local Shape of Offsets to Implicit Algebraic Curves. Mathematics in Computer Science 2(4), 635–652 (2008)
6. Cox, D., Little, J., O'Shea, D.: Ideals, Varieties, and Algorithms. Springer, New York (1997)
7. Farouki, R.T., Neff, C.A.: Analytic Properties of Plane Offset Curves. Computer Aided Geometric Design 7, 83–99 (1990)
8. Farouki, R.T., Neff, C.A.: Algebraic Properties of Plane Offset Curves. Computer Aided Geometric Design 7, 101–127 (1990)
9. Pottmann, H.: Rational Curves and Surfaces with Rational Offsets. Computer Aided Geometric Design 12, 175–192 (1995)
10. Sendra, J., Sendra, J.R.: Algebraic Analysis of Offsets to Hypersurfaces. Mathematische Zeitschrift 234, 697–719 (2000)
11. Sendra, J., Sendra, J.R.: An Algebraic Analysis of Conchoids to Algebraic Curves. Applicable Algebra in Engineering, Communication and Computing 19(5), 413–428 (2008)
12. Sendra, J., Sendra, J.R.: Rational Conchoids of Algebraic Curves. Applicable Algebra in Engineering, Communication and Computing 21, 285–308 (2010)
13. Walker, R.J.: Algebraic Curves. Princeton University Press, Princeton (1950)

Estimation of Integral Properties of a Planar Closed Curve Based on a Quadratic Spline Quasi-Interpolant

C. Allouch, P. Sablonnière, and D. Sbibih*

tafit0@hotmail.com,
Paul.Sablonniere@insa-rennes.fr,
sbibih@yahoo.fr

Abstract. In this paper, we present a method based on a quadratic spline quasi-interpolant for the estimation of integral properties of a planar closed curve. The latter include the length, area, center of gravity and moment of inertia of the given curve. Then, we analyze the error estimates on the approximations of these properties and we validate the theoretical results by numerical examples.

Keywords: Length, Area, Center of gravity, Spline quasi interpolant.

1 Introduction

Computing the arc length of a parametric curve has been well studied in the literature and has been treated in various ways. For example, in [7], the authors use numerical quadrature for estimating the length of the curve and in [6], a cubic spline interpolation based on the chord length parameterization. The method introduced in [7] is also used in [9] for estimating the area of a parametric surface. Let

$$\sigma : t \in [0,1] \to \mathbb{R}^2$$

$$\sigma(t) = (f(t), g(t))$$

be a parametric closed curve, by which we mean a continuously differentiable function such that $\sigma'(t) \neq 0$ for all $t \in [0,1]$. Denote by $|.|$ denote the Euclidian norm in $\mathbb{R}^2$. Then the length of σ (see [8]) is

$$\mathcal{L}(\sigma) := \int_0^1 |\sigma'(t)| dt = \int_0^1 \sqrt{f'(t)^2 + g'(t)^2} dt, \tag{1}$$

and its area is given by the formula

$$\mathcal{A}(\sigma) := \frac{1}{2} \int_0^1 (f(t)g'(t) - f'(t)g(t)) dt. \tag{2}$$

* Research supported by AI MA/08/182.

In this paper, we obtain approximations of $\mathcal{L}(\sigma)$ and $\mathcal{A}(\sigma)$ by replacing each component of σ by a quadratic spline quasi-interpolant and we compute the exact length and area of the spline approximant. We do the same for the center of gravity $\mathcal{G} = (x^*, y^*)$ of σ which has the coordinates

$$x^* := \frac{1}{\mathcal{L}(\sigma)} \int_0^1 f(t)|\sigma'(t)|dt, \quad y^* := \frac{1}{\mathcal{L}(\sigma)} \int_0^1 g(t)|\sigma'(t)|dt \tag{3}$$

and the moments of inertia with respect to an axis passing through $\mathcal{G}$ defined by

$$\mathcal{M}(\sigma) := \int_0^1 |\sigma(t) - \mathcal{G}|^2 |\sigma'(t)|dt. \tag{4}$$

We show that the orders of these approximations are that of the quadrature formula based on the used spline quasi-interpolant.

The paper has been arranged in the following way. In Section 2, we give the definition and the main properties of the used quadratic spline quasi-interpolant. In Section 3, we give details on the computation of the approximate length and center of gravity. We also study the error of the approximation. The approximations of the moment of inertia and the area are discussed in Sections 4 and 5. Numerical validation is given in Section 6.

2 Periodic Quadratic Spline Quasi-Interpolant

Let $\mathcal{X}_n = \{[x_i, x_{i+1}],\ 0 \le i \le n\}$ be the uniform partition of the interval $[0,1]$ with meshlength $h = \frac{1}{n}$; whenever necessary this partition will be extended periodically to $\mathbb{R}$. For $j = 1, 2, \ldots, n+2$, let $B_j = B(. - j)$ be the classical $\mathcal{C}^1$-quadratic B-spline of support $[x_{j-3}, x_j]$. The family $\{B_j,\ j = 1, 2, \ldots, n+2\}$ is a basis of the space $\mathcal{S}_2(I, \mathcal{X}_n)$ of $\mathcal{C}^1$ quadratic splines defined in I endowed with the partition $\mathcal{X}_n$. The $\mathcal{C}^1$ periodic quadratic spline quasi interpolant (abbr. QI) used here is the spline operator given for any f in $\mathcal{C}[0,1]$ by

$$\mathcal{Q}_n f := \sum_{j=1}^{n+2} \mu_j(f) B_j, \tag{5}$$

where

$$\mu_j(f) = \frac{1}{8}(-f_{j-2} + 10 f_{j-1} - f_j), \quad \text{for } 1 \le j \le n+2,$$

with $f_i = f(t_i)$ and $t_i = \left(i - \frac{1}{2}\right) h, \quad i = -1, \ldots, n+2.$

Theorem 1. *For all periodic function $f \in \mathcal{C}^3[0,1]$ we have (see [11])*

$$\|f^{(k)} - \mathcal{Q}_n f^{(k)}\|_\infty \le C_k \left(\frac{1}{n}\right)^{3-k} \|f^{(3-k)}\|_\infty, \quad k = 0, 1, 2, \tag{6}$$

where C_k is a constant independent of n. Moreover, for $f \in \mathcal{C}^4[0,1]$ we have the following superconvergent result

$$f(x_i) - \mathcal{Q}_n f(x_i) = \mathcal{O}\left(\frac{1}{n}\right)^4, \quad i = 0, \ldots, n. \tag{7}$$

Now, by integrating the quadratic QI $\mathcal{Q}_n$, we obtain the midpoint rule $M_n(f) = \int_0^1 \mathcal{Q}_n f(x)dx = \frac{1}{n}\sum_{i=1}^n f_i$. By using the symmetry of B-splines and the evaluation points t_i, one can prove the following result.

Theorem 2. *For a periodic function f of class $\mathcal{C}^4$ and a smooth weight function w we have (see [9])*

$$\mathcal{E}(f, w) = \int_0^1 [\mathcal{Q}_n f(x) - f(x)]w(x)dx < C_2 \left(\frac{1}{n}\right)^4, \tag{8}$$

where $C_2 > 0$ is a constant independent of n.

In the interval $[x_{i-1}, x_i]$, $i = 1, \ldots n$, by using the change of variable $x = (1-u)x_{i-1} + ux_i$, the QI $\mathcal{Q}_n$ becomes a quadratic polynomial in the variable $u \in [0,1]$ and it can be written as follows

$$\mathcal{Q}_n f = a_i \mathcal{B}_0 + b_i \mathcal{B}_1 + a_{i+1}\mathcal{B}_2, \tag{9}$$

where $\{\mathcal{B}_r = \binom{2}{r}(1-u)^{2-r}u^r,\ r = 0,1,2\}$ is the Bernstein basis of the space Π_2 of polynomials of degree 2 and

$$a_i = \frac{1}{16}\left(-f_{i-1} + 9f_i + 9f_{i+1} - f_{i+2}\right), \quad b_i = \frac{1}{8}\left(-f_{i-1} + 10f_i - f_{i+1}\right).$$

The representation (9) is used in the rest of the paper.

3 Length and Center of Gravity

Throughout this paper we denote $\bar{f}_n = \mathcal{Q}_n f$, $\bar{g}_n = \mathcal{Q}_n g$ and $\sigma_n(t) = (\bar{f}_n(t), \bar{g}_n(t))$. For $i = 1, \ldots, n$, let $\bar{f}_n|_{[x_{i-1},x_i]} = p_i$ and $\bar{g}_n|_{[x_{i-1},x_i]} = q_i$. From (9), the quadratic polynomials p_i and q_i can be written in the forms

$$p_i = a_i\mathcal{B}_0 + b_i\mathcal{B}_1 + a_{i+1}\mathcal{B}_2, \quad q_i = \alpha_i\mathcal{B}_0 + \beta_i\mathcal{B}_1 + \alpha_{i+1}\mathcal{B}_2.$$

We propose to approximate $\mathcal{L}(\sigma)$ by

$$\mathcal{L}(\sigma_n) = \int_0^1 |\sigma_n'(t)|dt = \int_0^1 \sqrt{\bar{f}_n'(t)^2 + \bar{g}_n'(t)^2}dt$$

and the center of gravity $\mathcal{G} = (x^*, y^*)$ by $\mathcal{G}_n = (x_n^*, y_n^*)$ where

$$x_n^* = \frac{1}{\mathcal{L}(\sigma_n)}\int_0^1 \bar{f}_n(t)|\sigma_n'(t)|dt, \quad y_n^* = \frac{1}{\mathcal{L}(\sigma_n)}\int_0^1 \bar{g}_n(t)|\sigma_n'(t)|dt.$$

In the next subsection, we give some details on the computation of the approximations of $\mathcal{L}(\sigma_n)$ and $\mathcal{G}_n$.

3.1 Approximations of $\mathcal{L}(\sigma)$ and $\mathcal{G}$

Let

$$\pi_i(u) = c_{0,i} + 2c_{1,i}u + c_{2,i}u^2, \quad i = 1, \dots, n,$$

where

$$\begin{cases} c_{0,i} = \gamma_i^2 + \bar{\gamma}_i^2 \\ c_{1,i} = 4[(\gamma_i - \delta_i)^2 + (\bar{\gamma}_i - \bar{\delta}_i)^2] \\ c_{2,i} = \gamma_i(\delta_i - \gamma_i) + \bar{\gamma}_i(\bar{\delta}_i - \bar{\gamma}_i) \end{cases}$$

and

$$\begin{cases} \gamma_i = b_i - a_i,\ \delta_i = a_{i+1} - b_i \\ \bar{\gamma}_i = \beta_i - \alpha_i, \bar{\delta}_i = \alpha_{i+1} - \beta_i. \end{cases}$$

We have the following result.

Theorem 3. *The approximate length and center of gravity are given by*

$$\begin{aligned} \mathcal{L}(\sigma_n) &= h\sum_{i=1}^{n} \mathcal{J}_{0,i} \\ x_n^* &= h\sum_{i=1}^{n} \left[a_i\mathcal{J}_{0,i} + 2(b_i - a_i)\mathcal{J}_{1,i} + (a_{i+1} - 2b_i + a_i)\mathcal{J}_{2,i}\right] \\ y_n^* &= h\sum_{i=1}^{n} \left[\alpha_i\mathcal{J}_{0,i} + 2(\beta_i - \alpha_i)\mathcal{J}_{1,i} + (\alpha_{i+1} - 2\beta_i + \alpha_i)\mathcal{J}_{2,i}\right]. \end{aligned}$$

where

$$\begin{aligned} \mathcal{J}_{0,i} &= \frac{c^2}{2a}\left[(t_1 - t_0) + \frac{1}{2}(\sinh(2t_1) - \sinh(2t_0))\right] \\ \mathcal{J}_{1,i} &= \frac{c^3}{3a^2}\left[\cosh(t_1)^3 - \cosh(t_0)^3\right] - \frac{b}{a}\mathcal{J}_{0,i} \\ \mathcal{J}_{2,i} &= \frac{c^4}{8a^3}\left[\frac{1}{4}(\sinh(4t_1) - \sinh(4t_0)) - (t_1 - t_0)\right] - \frac{2b}{a}\mathcal{J}_{1,i} - \frac{b^2}{a^2}\mathcal{J}_{0,i}. \end{aligned}$$

and

$$a = \sqrt{c_{2,i}}, \quad b = \frac{c_{1,i}}{\sqrt{c_{2,i}}}, \quad c = c_{0,i} - \frac{c_{1,i}^2}{\sqrt{c_{2,i}}}, \quad \sinh(t_0) = \frac{b}{c}, \quad \sinh(t_1) = \frac{a+b}{c}.$$

Proof. From (8), we obtain

$$\begin{aligned} \mathcal{L}(\sigma_n) &= \sum_{i=1}^{n}\int_{x_{i-1}}^{x_i} \sqrt{\bar{f}_n'(t)^2 + \bar{g}_n'(t)^2}dt \\ &= h\sum_{i=1}^{n}\int_0^1 \sqrt{p_i'(u)^2 + q_i'(u)^2}du \end{aligned}$$

and

$$x_n^* = h\frac{1}{\mathcal{L}(\sigma_n)}\sum_{i=1}^{n}\int_0^1 p_i(u)\sqrt{p_i'(u)^2+q_i'(u)^2}du,$$
$$y_n^* = h\frac{1}{\mathcal{L}(\sigma_n)}\sum_{i=1}^{n}\int_0^1 q_i(u)\sqrt{p_i'(u)^2+q_i'(u)^2}du.$$

Thus, we have to compute the following integrals

$$\int_0^1 p_i(u)\sqrt{p_i'(u)^2+q_i'(u)^2}du \quad \text{and} \quad \int_0^1 q_i(u)\sqrt{p_i'(u)^2+q_i'(u)^2}du$$

which is equivalent to evaluate integrals of type

$$\mathcal{J}_{0,i} := \int_0^1 \sqrt{\pi_i(u)}du, \quad \mathcal{J}_{1,i} := \int_0^1 u\sqrt{\pi_i(u)}du, \quad \mathcal{J}_{2,i} := \int_0^1 u^2\sqrt{\pi_i(u)}du,$$

where

$$\pi_i(u) := c_{0,i} + 2c_{1,i}u + c_{2,i}u^2 = \left(\sqrt{c_{2,i}}u + \frac{c_{1,i}}{\sqrt{c_{2,i}}}\right)^2 + \left(c_{0,i} - \frac{c_{1,i}^2}{c_{2,i}}\right).$$

For the sake of notational simplicity, the index i in $c_{k,i}$, $k = 0,1,2$ is dropped. It is easy to show that $c_1^2 - c_0c_2 < 0$, ($\pi_i(u)$ does not vanish), then by denoting $a := \sqrt{c_2}$, $b := \frac{c_1}{\sqrt{c_2}}$, $c := c_0 - \frac{c_1^2}{\sqrt{c_2}}$, we can write $\pi_i(u)$ as

$$\pi_i(u) = (au+b)^2 + c^2 = c^2\left[1 + \left(\frac{au+b}{c}\right)^2\right].$$

Setting $\phi(u) := 1 + (\frac{au+b}{c})^2$, we obtain

$$\mathcal{J}_{0,i} := c\int_0^1 \sqrt{\phi(u)}du, \quad \mathcal{J}_{1,i} := c\int_0^1 u\sqrt{\phi(u)}du, \quad \mathcal{J}_{2,i} := c\int_0^1 u^2\sqrt{\phi(u)}du.$$

Taking the change of variable

$$\sinh(t) = \frac{au+b}{c} \quad \text{where} \quad u = \frac{1}{a}(c\sinh(t) - b), \quad du = \frac{c}{a}\cosh(t)dt,$$

we define the new bounds t_0 and t_1 by

$$\sinh(t_0) := \frac{b}{c}, \quad \sinh(t_1) := \frac{a+b}{c}.$$

Then, we get $t_0 = \ln\left(\frac{b}{c} + \sqrt{1 + \left(\frac{b}{c}\right)^2}\right)$, $\quad t_1 = \ln\left(\frac{a+b}{c} + \sqrt{1 + \left(\frac{a+b}{c}\right)^2}\right)$ and $\cosh(t_0) = \sqrt{1 + b^2/c^2}$, $\quad \cosh(t_1) = \sqrt{1 + (a+b)^2/c^2}$.

The integrals become

$$\begin{aligned}\mathcal{J}_{0,i} &= \frac{c^2}{a}\int_{t_0}^{t_1}\sqrt{1+\sinh(t)^2}\cosh(t)dt = \frac{c^2}{a}\int_{t_0}^{t_1}\cosh(t)^2dt \\ &= \frac{c^2}{2a}\int_{t_0}^{t_1}(1+\cosh(2t))dt = \frac{c^2}{2a}\left[(t_1-t_0)+\frac{1}{2}(\sinh(2t_1)-\sinh(2t_0))\right]\end{aligned}$$

$$\begin{aligned}\mathcal{J}_{1,i} &= \frac{c^2}{a^2}\int_{t_0}^{t_1}(c\sinh(t)-b)\cosh(t)^2dt = \frac{c^3}{a^2}\int_{t_0}^{t_1}\cosh(t)^2\sinh(t)dt - \frac{b}{a}\mathcal{J}_{0,i} \\ &= \frac{c^3}{3a^2}\left[\cosh(t_1)^3-\cosh(t_0)^3\right] - \frac{b}{a}\mathcal{J}_{0,i}\end{aligned}$$

$$\begin{aligned}\mathcal{J}_{2,i} &= \frac{c^2}{a^3}\int_{t_0}^{t_1}(c\sinh(t)-b)^2\cosh(t)^2dt \\ &= \frac{c^4}{a^3}\int_{t_0}^{t_1}\sinh(t)^2\cosh(t)^2dt - \frac{2b}{a}\mathcal{J}_{1,i} - \frac{b^2}{a^2}\mathcal{J}_{0,i} \\ &= \frac{c^4}{8a^3}\int_{t_0}^{t_1}(\cosh(4t)-1)dt - \frac{2b}{a}\mathcal{J}_{1,i} - \frac{b^2}{a^2}\mathcal{J}_{0,i} \\ &= \frac{c^4}{8a^3}\left[\frac{1}{4}(\sinh(4t_1)-\sinh(4t_0))-(t_1-t_0)\right] - \frac{2b}{a}\mathcal{J}_1 - \frac{b^2}{a^2}\mathcal{J}_{0,i}.\end{aligned}$$

Hence, we obtain

$$\begin{aligned}\int_0^1 p_i(u)\sqrt{\pi_i(u)}du &= a_i\int_0^1\sqrt{\pi_i(u)}du + 2(b_i-a_i)\int_0^1 u\sqrt{\pi_i(u)}du \\ &\quad + (a_{i+1}-2b_i+a_i)\int_0^1 u^2\sqrt{\pi_i(u)}du \\ &= a_i\mathcal{J}_{0,i} + 2(b_i-a_i)\mathcal{J}_{1,i} + (a_{i+1}-2b_i+a_i)\mathcal{J}_{2,i}\end{aligned}$$

and similarly

$$\begin{aligned}\int_0^1 q_i(u)\sqrt{\pi_i(u)}du &= \alpha_i\int_0^1\sqrt{\pi_i(u)}du + 2(\beta_i-\alpha_i)\int_0^1 u\sqrt{\pi_i(u)}du \\ &\quad + (\alpha_{i+1}-2\beta_i+\alpha_i)\int_0^1 u^2\sqrt{\pi_i(u)}du \\ &= \alpha_i\mathcal{J}_{0,i} + 2(\beta_i-\alpha_i)\mathcal{J}_{1,i} + (\alpha_{i+1}-2\beta_i+\alpha_i)\mathcal{J}_{2,i},\end{aligned}$$

which completes the proof.

3.2 Convergence Orders

Theorem 4. *For $\sigma \in \mathcal{C}^4[0,1]$, we have*

$$|\mathcal{L}(\sigma)-\mathcal{L}(\sigma_n)| = \mathcal{O}(h^4), \quad \textit{as} \quad h \to 0 \tag{10}$$

and

$$|\mathcal{G}-\mathcal{G}_n| = \mathcal{O}(h^4), \quad \textit{as} \quad h \to 0 \tag{11}$$

where $|\mathcal{G}-\mathcal{G}_n| = \sqrt{(x_{\mathcal{G}}-x_n^*)^2+(y_{\mathcal{G}}-y_n^*)^2}$ *and* $h=\frac{1}{n}$.

Proof. Let $e(t)=\sigma(t)-\sigma_n(t)$. From (6) we have

$$\|e\| = \mathcal{O}(h^3), \quad \|e'\| = \mathcal{O}(h^2), \quad \text{as} \quad h \to 0. \tag{12}$$

Let us use the identity

$$|\sigma_n'|-|\sigma'| = \frac{-2e'.\sigma'+e'.e'}{|\sigma_n'|+|\sigma'|}$$

and the fact that $|\sigma'|$ and $|\sigma_n'|$ are bounded away from zero for small enough h, by the bound on e' in (12), we so obtain

$$\mathcal{L}(\sigma_n)-\mathcal{L}(\sigma) = \int_0^1 \left[|\sigma_n'(t)|-|\sigma'(t)|\right]dt = -2\int_0^1 \frac{e'(t).\sigma'(t)}{|\sigma_n'(t)|+|\sigma'(t)|}dt + \mathcal{O}(h^4).$$

But since $e(0)=e(1)=\mathcal{O}(h^4)$, an integration by parts implies

$$-\int_0^1 \frac{e'(t).\sigma'(t)}{|\sigma_n'(t)|+|\sigma'(t)|}dt = \int_0^1 e(t).\frac{d}{dt}\left(\frac{\sigma'(t)}{|\sigma_n'(t)|+|\sigma'(t)|}\right)dt + \mathcal{O}(h^4).$$

Since $|\sigma'(t)|, |\sigma''(t)|, |\sigma_n'(t)|$ and $|\sigma_n''(t)|$ are bounded as $h \to 0$, so too is

$$\frac{d}{dt}\left(\frac{\sigma'(t)}{|\sigma_n'(t)|+|\sigma'(t)|}\right)$$

and the estimate (10) follows from the bound in (8). Now we write

$$\begin{aligned} x^*-x_n^* &= \frac{1}{\mathcal{L}(\sigma)}\int_0^1 f(t)|\sigma'(t)|dt - \frac{1}{\mathcal{L}(\sigma_n)}\int_0^1 f(t)|\sigma_n'(t)|dt \\ &= \left(\frac{1}{\mathcal{L}(\sigma)}-\frac{1}{\mathcal{L}(\sigma_n)}\right)\int_0^1 f(t)|\sigma'(t)|dt \\ &\qquad + \frac{1}{\mathcal{L}(\sigma_n)}\left(\int_0^1 (\bar{f}_n(t)|\sigma'(t)|-\bar{f}_n(t)|\sigma_n'(t)|)dt\right). \end{aligned} \tag{13}$$

From (10), we have

$$\begin{aligned} &\left(\frac{1}{\mathcal{L}(\sigma)}-\frac{1}{\mathcal{L}(\sigma_n)}\right)\int_0^1 f(t)|\sigma'(t)|dt \\ &\qquad = \left(\frac{\mathcal{L}(\sigma_n)-\mathcal{L}(\sigma)}{\mathcal{L}(\sigma_n)\mathcal{L}(\sigma)}\right)\int_0^1 f(t)|\sigma'(t)|dt = \mathcal{O}(h^4). \end{aligned} \tag{14}$$

On the other hand

$$\frac{1}{\mathcal{L}(\sigma_n)}\int_0^1 \left(\bar{f}_n|\sigma'|-\bar{f}_n|\sigma_n'|\right) = \frac{1}{\mathcal{L}(\sigma_n)}\int_0^1 \left(f(|\sigma'|-|\sigma_n'|)+(f-\bar{f}_n)|\sigma_n'|\right)$$

we proceeded as for (10) to show that

$$\int_0^1 f(t)(|\sigma'(t)| - |\sigma_n'(t)|) = \mathcal{O}(h^4). \tag{15}$$

Moreover, using (8) we have

$$\int_0^1 (f(t) - \bar{f}_n(t))|\sigma_n'(t)|dt = \mathcal{E}(f, |\sigma_n'(t)|) = \mathcal{O}(h^4). \tag{16}$$

Now combining the estimates (13)-(16), we deduce that $|x^* - x_n^*| = \mathcal{O}(h^4)$. Similarly, we prove that $|y^* - y_n^*| = \mathcal{O}(h^4)$, and the estimate (11) follows.

4 Moment of Inertia

4.1 Approximate Moment of Inertia

By using the same notations as before, we have the following result.

Theorem 5. *Let*

$$\begin{cases} \bar{a}_i = a_i - x_n^*, \ \bar{b}_i = b_i - y_n^*, \\ \bar{\alpha}_i = \alpha_i - x_n^*, \ \bar{\beta}_i = \beta_i - y_n^*. \end{cases}$$

The approximate moment of inertia is given by

$$\mathcal{M}(\sigma_n) = h \sum_{i=1}^{n} \sum_{k=0}^{4} d_k \mathcal{J}_{k,i},$$

where

$$\mathcal{J}_{3,i} = \frac{c^5}{a^4} \int_{t_0}^{t_1} \sinh(t)^3 \cosh(t)^2 dt - \frac{3b}{a}\mathcal{J}_{2,i} - \frac{3b^2}{a^2}\mathcal{J}_{1,i} - \frac{b^3}{a^3}\mathcal{J}_{0,i},$$
$$\mathcal{J}_{4,i} = \frac{c^6}{a^5} \int_{t_0}^{t_1} \sinh(t)^4 \cosh(t)^2 dt - \frac{4b}{a}\mathcal{J}_{3,i} - \frac{6b^2}{a^2}\mathcal{J}_{2,i} - \frac{4b^3}{a^3}\mathcal{J}_{1,i} - \frac{b^4}{a^4}\mathcal{J}_{0,i},$$

and $\{d_k, \ 0 \le k \le 4\}$ *are the canonical basis coefficients of the polynomial*

$$k_i(u) = (\bar{a}_i(1-u)^2 + 2\bar{b}_i u(1-u) + \bar{a}_{i+1}u^2)^2 + (\bar{\alpha}_i(1-u)^2 + 2\bar{\beta}_i u(1-u) + \bar{\alpha}_{i+1}u^2)^2.$$

Proof. Since

$$\begin{aligned} \mathcal{M}(\sigma_n) &= \int_0^1 |\sigma_n(t) - \mathcal{G}_n|^2 |\sigma_n'(t)| dt \\ &= \int_0^1 \left[(\bar{f}_n(t) - x_n^*)^2 + (\bar{g}_n(t) - y_n^*)^2\right] \sqrt{\bar{f}_n'(t)^2 + \bar{g}_n'(t)^2} dt \\ &= h \sum_{i=1}^{n} \int_0^1 \left[(p_i(u) - x_n^*)^2 + (q_i(u) - y_n^*)^2\right] \sqrt{\pi_i(u)} du, \end{aligned}$$

then, we have to compute the integrals

$$\int_0^1 \left[(p_i(u)-x_n^*)^2+(q_i(u)-y_n^*)^2\right]\sqrt{\pi_i(u)}du, \quad i=1,\ldots,n. \tag{17}$$

The expression between brackets can be written

$$k_i(u):=(\bar{a}_i(1-u)^2+2\bar{b}_iu(1-u)+\bar{a}_{i+1}u^2)^2+(\bar{\alpha}_i(1-u)^2+2\bar{\beta}_iu(1-u)+\bar{\alpha}_{i+1}u^2)^2 \tag{18}$$

or simply

$$k_i(u)=d_0+d_1u+d_2u^2+d_3u^3+d_4u^4.$$

Thus, the integrals given by (17) become

$$\sum_{k=0}^{4} d_k\mathcal{J}_{k,i}, \quad \text{with} \quad \mathcal{J}_{k,i}:=\int_0^1 u^k\sqrt{\pi_i(u)}du.$$

Hence, we have to compute the new integrals

$$\mathcal{J}_{3,i}:=\int_0^1 u^3\sqrt{\pi_i(u)}du=\frac{c^2}{a^4}\int_{t_0}^{t_1}(c\sinh(t)-b)^3\cosh(t)^2dt$$

$$\mathcal{J}_{4,i}:=\int_0^1 u^4\sqrt{\pi_i(u)}du=\frac{c^2}{a^5}\int_{t_0}^{t_1}(c\sinh(t)-b)^4\cosh(t)^2dt.$$

By using the expression of the integrals $\mathcal{J}_{k,i},\ k=0,1,2$, we get

$$\begin{aligned}\mathcal{J}_{3,i}:&=\frac{c^5}{a^4}\int_{t_0}^{t_1}\sinh(t)^3\cosh(t)^2dt-\frac{3bc^4}{a^4}\int_{t_0}^{t_1}\sinh(t)^2\cosh(t)^2dt\\&\quad+\frac{3b^2c^3}{a^4}\int_{t_0}^{t_1}\sinh(t)\cosh(t)^2dt-\frac{b^3c^2}{a^4}\int_{t_0}^{t_1}\cosh(t)^2dt\\&=\frac{c^5}{a^4}\int_{t_0}^{t_1}\sinh(t)^3\cosh(t)^2dt-\frac{3b}{a}\mathcal{J}_{2,i}-\frac{3b^2}{a^2}\mathcal{J}_{1,i}-\frac{b^3}{a^3}\mathcal{J}_{0,i}\end{aligned}$$

and similarly

$$\begin{aligned}\mathcal{J}_{4,i}:&=\frac{c^6}{a^5}\int_{t_0}^{t_1}\sinh(t)^4\cosh(t)^2dt-\frac{4b}{a}\left(\mathcal{J}_{3,i}+\frac{3b}{a}\mathcal{J}_{2,i}+\frac{3b^2}{a^2}\mathcal{J}_{1,i}+\frac{b^3}{a^3}\mathcal{J}_{0,i}\right)\\&\quad+\frac{6b^2}{a^2}\left(\mathcal{J}_{2,i}+\frac{2b}{a}\mathcal{J}_{1,i}+\frac{b^2}{a^2}\mathcal{J}_{0,i}\right)-\frac{4b^3}{a^3}\left(\mathcal{J}_{1,i}+\frac{b}{a}\mathcal{J}_{0,i}\right)+\frac{b^4}{a^4}\mathcal{J}_{0,i}\\&=\frac{c^6}{a^5}\int_{t_0}^{t_1}\sinh(t)^4\cosh(t)^2dt-\frac{4b}{a}\mathcal{J}_{3,i}-\frac{6b^2}{a^2}\mathcal{J}_{2,i}-\frac{4b^3}{a^3}\mathcal{J}_{1,i}-\frac{b^4}{a^4}\mathcal{J}_{0,i}.\end{aligned}$$

Now, by linearizing

$$\sinh(t)^3\cosh(t)^2=\frac{1}{16}(\sinh(5t)-2\sinh(t)-\sinh(3t))$$

$$\sinh(t)^4\cosh(t)^2=\frac{1}{32}(2-\cosh(2t)-2\cosh(4t)+\cosh(6t))$$

we deduce that

$$\begin{aligned}\int_{t_0}^{t_1} \sinh(t)^3 \cosh(t)^2 dt &= \left[\frac{1}{80}\cosh(5t) - \frac{1}{8}\cosh(t) - \frac{1}{48}\cosh(3t)\right] \\ &= \left[\frac{1}{5}\cosh(t)^5 - \frac{1}{3}\cosh(t)^3\right]_{t_0}^{t_1}\end{aligned}$$

$$\begin{aligned}\int_{t_0}^{t_1} \sinh(t)^4 \cosh(t)^2 dt &= \left[\frac{1}{16}t - \frac{1}{64}\sinh(2t) - \frac{1}{64}\sinh(4t) + \frac{1}{192}\sinh(6t)\right]_{t_0}^{t_1} \\ &= \left[\frac{1}{16}t + \sinh(t)\cosh(t)\left(\frac{1}{16} - \frac{7}{24}\cosh(t)^2 + \frac{1}{6}\cosh(t)^4\right)\right]_{t_0}^{t_1}\end{aligned}$$

which completes the computation of $\mathcal{J}_{3,i}$ and $\mathcal{J}_{4,i}$.

4.2 Convergence Order

Theorem 6. *For $\sigma \in \mathcal{C}^4[0,1]$, we have*

$$|\mathcal{M}(\sigma) - \mathcal{M}(\sigma_n)| = \mathcal{O}(h^4), \quad as \quad h \to 0. \tag{19}$$

Proof. Writing

$$\begin{aligned}\mathcal{M}(\sigma) - \mathcal{M}(\sigma_n) &= \int_0^1 |\sigma(t) - \mathcal{G}|^2 |\sigma'(t)| dt - \int_0^1 |\sigma_n(t) - \mathcal{G}_n|^2 |\sigma_n'(t)| dt \\ &= \int_0^1 |\sigma(t) - \mathcal{G}|^2 (|\sigma'(t)| - |\sigma_n'(t)|) dt \\ &+ \int_0^1 |\sigma_n'(t)| \left(|\sigma(t) - \mathcal{G}|^2 - |\sigma_n(t) - \mathcal{G}_n|^2\right) dt \\ &= C_n + D_n\end{aligned}$$

and

$$\begin{aligned}D_n = \int_0^1 |\sigma_n'(t)| [&(f - \bar{f}_n + x_n^* - x^*)(f + \bar{f}_n - x^* - x_n^*) \\ &+ (g - \bar{g}_n + y_n^* - y^*)(g + \bar{g}_n - y^* - y_n^*)] dt.\end{aligned}$$

Since $|x^* - x_n^*| = \mathcal{O}(h^4)$ and $|y^* - y_n^*| = \mathcal{O}(h^4)$, we deduce from (8) that $D_n = \mathcal{O}(h^4)$. We proceeded as for (10) to show that $C_n = \mathcal{O}(h^4)$, then the result follows.

5 Area

5.1 Approximate Area

Theorem 7. *The approximate area is given by*

$$\mathcal{A}(\sigma_n) = \frac{h}{6}\sum_{i=1}^{n} [2(a_i\beta_i - b_i\alpha_i) + (a_i\alpha_{i+1} - a_{i+1}\alpha_i) + 2(b_i\alpha_{i+1} - a_{i+1}\beta_i)].$$

Proof. We approximate $\mathcal{A}(\sigma)$ by

$$\mathcal{A}(\sigma_n) := \frac{1}{2}\int_0^1 (\bar{f}_n(t)\bar{g}_n'(t) - \bar{f}_n'(t)\bar{g}_n(t))dt.$$

Thus, we have to compute the integrals

$$\frac{1}{2}\int_0^1 (p_i(u)q_i'(u) - p_i'(u)q_i(u))du, \quad i = 1, \ldots, n$$

with

$$p_i' = 2[\gamma_i(1-u) + \delta_i u], \quad q_i' = 2[\bar{\gamma}_i(1-u) + \bar{\delta}_i u]$$

and

$$\begin{cases} \gamma_i = b_i - a_i, \; \delta_i = a_{i+1} - b_i \\ \bar{\gamma}_i = \beta_i - \alpha_i, \bar{\delta}_i = \alpha_{i+1} - \beta_i. \end{cases}$$

Then, we obtain

$$\begin{aligned} \frac{1}{2}(p_i q_i' - p_i' q_i) &= (a_i(1-u)^2 + 2b_i u(1-u) + a_{i+1}u^2)(\bar{\gamma}_i(1-u) + \bar{\delta}_i u) \\ &\quad - (\gamma_i(1-u) + \delta_i u)(\alpha_i(1-u)^2 + 2\beta_i u(1-u) + \alpha_{i+1}u^2) \\ &= (a_i\bar{\gamma}_i - \alpha_i\gamma_i)(1-u)^3 + (a_i\bar{\delta}_i + 2b_i\bar{\gamma}_i - \alpha_i\delta_i - 2\alpha_{i+1}\gamma_i)u(1-u)^2 \\ &\quad + (2b_i\delta_i + a_{i+1}\bar{\gamma}_i - 2\beta_i\delta_i - \alpha_{i+1}\gamma_i)u^2(1-u) + (a_{i+1}\bar{\delta}_i - \alpha_{i+1}\delta_i)u^3 \\ &= (a_i\beta_i - b_i\alpha_i)(1-u)^3 + (a_i(\beta_i + \alpha_{i+1}) - (b_i + a_{i+1})\alpha_i)u(1-u)^2 \\ &\quad + ((\alpha_{i+1}(a_i + b_i) - (\alpha_i + \beta_i)a_{i+1}))u^2(1-u) + (b_i\alpha_{i+1} - a_{i+1}\beta_i)u^3 \end{aligned}$$

and consequently

$$\begin{aligned} \frac{1}{2}\int_0^1 (p_i q_i' - p_i' q_i) &= \frac{1}{12}(3a_i\beta_i - 3b_i\alpha_i + a_i\beta_i + a_i\alpha_{i+1} - b_i\alpha_i - a_{i+1}\alpha_i) \\ &\quad + \frac{1}{12}(a_i\alpha_{i+1} + b_i\alpha_{i+1} - a_{i+1}\alpha_i - a_{i+1}\beta_i + 3b_i\alpha_{i+1} - 3a_{i+1}\beta_i) \\ &= \frac{1}{6}(2(a_i\beta_i - b_i\alpha_i) + (a_i\alpha_{i+1} - a_{i+1}\alpha_i) + 2(b_i\alpha_{i+1} - a_{i+1}\beta_i)), \end{aligned}$$

which completes the proof. □

5.2 Convergence Order

Theorem 8. *For $\sigma \in \mathcal{C}^4[0,1]$, we have*

$$|\mathcal{A}(\sigma) - \mathcal{A}(\sigma_n)| = \mathcal{O}(h^4), \quad \textit{as} \quad h \to 0. \tag{20}$$

Proof. Writing

$$fg' - f'g - \bar{f}_n\bar{g}_n' + \bar{f}_n'\bar{g}_n = f(g' - \bar{g}_n') + \bar{g}_n'(f - \bar{f}_n) - f'(g - \bar{g}_n) - \bar{g}_n(f' - \bar{f}_n')$$

and using the fact that $f(0) = \mathcal{Q}_n f(0)$ and $f(1) = \mathcal{Q}_n f(1)$, then an integration by parts gives

$$\int_0^1 f(g' - \bar{g}_n') = -\int_0^1 f'(g - \bar{g}_n) \quad \text{and} \quad \int_0^1 \bar{g}_n(\bar{f}_n' - f') = -\int_0^1 \bar{g}_n'(f_n - f).$$

Hence

$$\mathcal{A}(\sigma) - \mathcal{A}(\sigma_n) = -2\int_0^1 f'(g - \bar{g}_n) - 2\int_0^1 \bar{g}_n'(f - \bar{f}_n) = -2\mathcal{E}(g, f') - 2\mathcal{E}(f, g')$$

and consequently (20) follows from (8). □

6 Numerical Results

We consider the following closed curves:

(**Raphal Laporte heart** [8]**):** $\sigma_1(s) = (\sin^3(s), \cos(s) - \cos^4(s))$.

(**Cassini oval** [1]**):** $\sigma_2(s) = R(s)(\cos(s), b\sin(s))$, where

$$R(s) = \sqrt{\cos(2s) + \sqrt{a - \sin^2(2s)}}.$$

(**Amoeba** [3]**):** $\sigma_3(s) = R(s)(\cos(s), \sin(s))$, where $R(s) = e^{\cos(s)}\cos^2(2s) + e^{\sin(s)}\sin^2(2s)$.

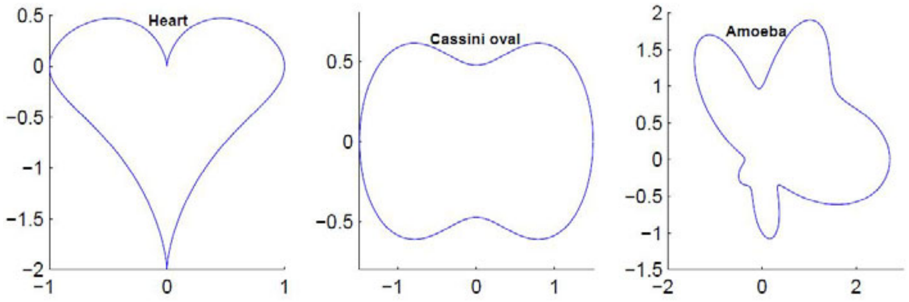


Fig. 1. Several closed curves

In Tables 1-5 for different values of n we give the approximation errors obtained by the estimation of integral properties of the curves $\sigma_i,\ i = 1, 2, 3$ using the QI $\mathcal{Q}_n$. We give also the numerical convergence orders of the approximations.

Table 1. Arc length approximation with $a = \frac{3}{2}$ and $b = 1$

n	—$\mathcal{L}(\sigma_1) - \mathcal{L}(\sigma_{1,n})$—		—$\mathcal{L}(\sigma_2) - \mathcal{L}(\sigma_{2,n})$—		—$\mathcal{L}(\sigma_3) - \mathcal{L}(\sigma_{3,n})$—	
8	1.17(-00)	-	5.57(-01)	-	4.32(-00)	-
16	1.33(-01)	3.13	7.85(-02)	2.83	1.30(-00)	1.73
32	9.60(-03)	3.79	6.24(-03)	3.65	1.40(-01)	3.22
64	6.14(-04)	3.97	4.16(-04)	3.90	9.80(-03)	3.84
128	3.82(-05)	4.01	2.64(-05)	3.98	6.27(-04)	3.97
256	2.35(-06)	4.02	1.66(-06)	3.99	3.97(-05)	3.98
512	1.45(-07)	4.02	1.04(-08)	4.00	2.50(-06)	3.99

Table 2. Approximation of the center of gravity

n	$\lvert\mathcal{G} - \mathcal{G}_n\rvert_{(\sigma_1)}$		$\lvert\mathcal{G} - \mathcal{G}_n\rvert_{(\sigma_2)}$		$\lvert\mathcal{G} - \mathcal{G}_n\rvert_{(\sigma_3)}$	
8	2.71(-02)	-	0.	-	4.89(-02)	-
16	2.13(-03)	3.67	0.	-	7.93(-04)	5.95
32	1.72(-04)	3.63	0.	-	1.52(-03)	0.94
64	1.19(-05)	3.85	0.	-	1.44(-04)	3.40
128	7.79(-07)	3.93	0.	-	1.02(-05)	3.83
256	5.04(-08)	3.95	0.	-	6.35(-07)	4.00
512	3.27(-09)	3.95	0.	-	3.90(-08)	4.03

Table 3. Approximation of the center of gravity

n	$\lvert\mathcal{G} - \mathcal{G}_n\rvert_{(\sigma_2)}$	
7	1.24(-02)	-
15	3.27(-04)	4.76
31	4.91(-07)	8.96
63	3.69(-12)	16.25
127	3.24(-16)	12.86

Table 4. Approximation of the moment of inertia

n	—$\mathcal{M}(\sigma_1) - \mathcal{M}(\sigma_{1,n})$—		—$\mathcal{M}(\sigma_2) - \mathcal{M}(\sigma_{2,n})$—		—$\mathcal{L}(\sigma_3) - \mathcal{M}(\sigma_{3,n})$—	
8	1.52(-00)	-	1.79(-00)	-	14.19(-00)	-
16	1.72(-01)	3.14	2.51(-01)	2.84	4.58(-00)	1.63
32	1.23(-02)	3.80	1.87(-02)	3.75	5.07(-01)	3.17
64	7.88(-04)	3.97	1.21(-03)	3.94	3.61(-02)	3.81
128	4.91(-05)	4.01	7.66(-05)	3.99	2.33(-03)	3.96
256	3.04(-06)	4.02	4.80(-06)	4.00	1.47(-04)	3.99
512	1.87(-07)	4.02	3.00(-07)	4.00	9.20(-06)	3.99

Remark 1. *It can be seen from the above tables, that the orders of convergence agree with the theoretical results. However we see that we obtain an exactness for the curve σ_2 for even values of n. This result can be proved by using the symmetry of the curve and the periodicity of the QI.*

Table 5. Area approximation with $a = \frac{3}{2}$ and $b = 1$

n	—$\mathcal{A}(\sigma_1) - \mathcal{A}(\sigma_{1,n})$—		—$\mathcal{A}(\sigma_2) - \mathcal{A}(\sigma_{2,n})$—		—$\mathcal{A}(\sigma_3) - \mathcal{A}(\sigma_{3,n})$—	
8	3.21(-02)	-	3.70(-01)	-	8.04(-01)	-
16	2.12(-03)	3.92	4.88(-02)	2.92	4.98(-01)	0.69
32	1.34(-04)	3.98	3.81(-03)	3.68	5.72(-02)	3.12
64	8.43(-06)	4.00	2.52(-04)	3.92	4.11(-03)	3.80
128	5.27(-07)	4.00	1.59(-05)	3.98	2.66(-04)	3.95
256	3.30(-08)	4.00	1.00(-06)	4.00	1.67(-05)	3.99
512	2.06(-09)	4.00	6.26(-08)	4.00	1.05(-06)	4.00

References

1. Brauer, A.: Limits for the characteristic roots of a matrix II. Duke Math. J. 14, 21–26 (1947)
2. Davis, P.J., Vitale, R.A., Ben-Sabar, E.: On the deterministic and stochastic approximation of regions. J. Approx. Theory 21(1), 60–88 (1977)
3. Bailey, D.H., Crandall, R.E.: Experimental Mathematics 11(4), 527–546 (2002)
4. Dubuc, S., Merrien, J.-L., Sablonnière, P.: The length of the de Rham curve. J. Math. Anal. Appl. 223, 182–195 (1998)
5. Floater, M.: Arc length estimation and the convergence of parametric polynomial interpolation. Preprint CMA, Oslo (2005)
6. Floater, M.: Chordal cubic spline interpolation is fourth order accurate. IMA J. Numer. Anal. 26, 25–33 (2006)
7. Floater, M., Rasmussen, A.F.: Point-based methods for estimating the length of a parametric curve. J. Comput. Applied Math. 196, 512–522 (2006)
8. Kreyszig, E.: Differential Geometry. Dover, New York (1991)
9. Rasmussen, A.F., Floater, M.S.: A point-based method for estimating surface area. In: Field, D., Gonsor, D., Neamtu, M. (eds.) Electronic Proceedings of the SIAM Conference on Geometric Design, Phoenix (2005)
10. Sablonnière, P.: Comparison of some approximation methods for computing arc lengths. Congrés NTA (Nouvelles Tendances en Approximation). University of Oujda (October 2009)
11. Sablonnière, P.: Univariate spline quasi-interpolants and applications to numerical analysis. Rend. Sem. Mat. Univ. Pol. Torino 63(2), 107–118 (2005)

Design of Multiresolution Operators Using Statistical Learning Tools: Application to Compression of Signals

Francesc Aràndiga[1], Albert Cohen[2], and Dionisio F. Yáñez[3]

[1] Dept. Matemática Aplicada. Universitat de València, C/Doctor Moliner, 46100 Burjassot (Valencia), Spain
arandiga@uv.es
http://www.uv.es/arandiga

[2] Laboratoire Jacques-Louis Lions. Universit Pierre et Marie Curie. 4, Place Jussieu 75005 Paris, France
cohen@ann.jussieu.fr
http://www.ann.jussieu.fr/~cohen

[3] Dept. Matemáticas, CC. Naturales y CC. Sociales aplicadas a la Educación, U. Católica de Valencia. C/Sagrado Corazón, 46103 Godella (Valencia), Spain
dionisiofelix.yanez@ucv.es
http://www.uv.es/diofeya

Abstract. Using multiresolution based on Harten's framework [*J. Appl. Numer. Math.*, 12 (1993), pp. 153–192.] we introduce an alternative to construct a prediction operator using Learning statistical theory. This integrates two ideas: generalized wavelets and learning methods, and opens several possibilities in the compressed signal context. We obtain theoretical results which prove that this type of schemes (LMR schemes) are equal to or better than the classical schemes. Finally, we compare traditional methods with the algorithm that we present in this paper.

Keywords: Learning-based multiresolution, Multiscale decomposition, Signal processing.

1 Introduction and Review

Multiscale representations of signals and images into wavelet bases have been successfully used in applications such as compression and denoising. In these applications, one essentially takes advantage of the *sparsity* of the representation of the image (see, e.g., [6,9,16]).

Harten [15] integrates ideas from three different fields such as theory of wavelets, numerical solution of partial differential equations (PDEs), and subdivision schemes. The aim of this paper is to incorporate the Learning theory (see, e.g., [10,17]) into the general framework for multiresolution of data and to use these results to data compression.

Based on multigrid methods (which typically use discretization by point-value and reconstruction by interpolation), Harten developed the idea that if we consider a sequence of grids with corresponding discretization $\{\mathcal{D}_k\}$ and reconstruction $\{\mathcal{R}_k\}$, then the most natural way to go from the kth grid to the coarser $(k-1)$th grid is by the operator $\mathcal{D}_k^{k-1} = \mathcal{D}_{k-1}\mathcal{R}_k$ and similarly to use the operator $\mathcal{P}_{k-1}^k = \mathcal{D}_k\mathcal{R}_{k-1}$ to go from $(k-1)$th grid to the finer kth grid.

Then we consider the notion of nested discretization in a more abstract setting and observe that the operators $\mathcal{D}_k^{k-1}$ and $\mathcal{P}_{k-1}^k$ can serve, respectively, as decimation and prediction in a pyramid scheme of the type that is used in signal processing. Using ideas from the theory of wavelet, we remove the redundancy that is typical of frames achieved by the use of pyramid schemes and we obtain a multiresolution representation. Furthermore, Harten used knowledge from the theory of wavelets to relate the discrete multiresolution representation to a multiresolution basis in the space of functions $\mathcal{F}$ (see, e.g., [8,7,16]). He shows that the construction of wavelets (see, e.g., [8]) can be formulated in terms of discretization and reconstruction (corresponding to a nested dyadic sequence of uniform grids). For this, the operator reconstruction may or may not be a *linear* operator. This is Harten's main contribution.

At this point we observe ([4,14,15]) that data compression based on multiresolution (as in Harten's) uses the fact that one discrete space V^k is divided in two spaces V^{k-1} and W^{k-1}, i.e.,

$$V^k = V^{k-1} \oplus W^{k-1},$$

where $V^{k-1} = \mathcal{D}_k^{k-1}(V^k)$ and W^{k-1} represents non redundant information present in V^k and not predictable from V^{k-1} by the prediction operator $\mathcal{P}_{k-1}^k$, i.e., $W^{k-1} = V^k - \mathcal{P}_{k-1}^k(V^{k-1})$. So, if the prediction operator is a "good approximation" the details are close to zero and the data can be compressed better. Therefore, Harten (see, e.g., [14,15]), and Aràndiga and Donat [4] emphasize the importance to construct a good reconstruction operator. They formulate the problem as a typical problem in approximation theory:

$$\text{Find a “good approximation” to } V^k \text{ from } V^{k-1}.$$

However, we observe that we know the space V^k, that is we have all the information of the next step. Why don't we use this information to obtain a prediction operator? So, we can formulate the problem as a typical problem in Learning theory (see, e.g., [10,17]):

$$\text{Knowing } V^{k-1} \text{ and } V^k, \text{ find a “good approximation” to } V^k \text{ from } V^{k-1}.$$

This work is organized as follows: In §2 we review the Multiresolution "à la Harten", then we introduce Learning-based multiresolution based on the minimization of a functional (see [2,18]). We develop this technique in the theoretical sense in §3, in particular in discretization for point-value context and explain the Learning process in MR context with an extensive example. We review LMR methods using the minimization of ℓ^p norm. Finally we do some numerical experiments comparing with classical linear methods and obtaining relevant results in §4.

2 Learning-Based Multiresolution in Harten's Framework

2.1 Generalized Wavelets. Brief Review

In Harten's framework there are two principal operators: the discretization operator, $\mathcal{D}_k$, that assigns to any function f a sequence of real numbers f^k; and the reconstruction operator, $\mathcal{R}_k$, that for any sequence of real numbers f^k obtains a function $\mathcal{R}_k f^k$. We find in the literature some references about the design of $\mathcal{D}_k$ (see e.g. [3,11,14,15]) but the main issue in Harten's MR literature is to obtain a good reconstruction operator using linear and nonlinear interpolation techniques (see e.g. [4,1,3]).

Combining these operators we obtain two other operators: The decimation operator $\mathcal{D}_k^{k-1} = \mathcal{D}_{k-1}\mathcal{R}_k$ is a linear operator that yields the discrete information contents of the signal at the resolution $k-1$ from the discrete information at level k (an increasing k implies more resolution). The prediction operator, $\mathcal{P}_{k-1}^k = \mathcal{D}_k\mathcal{R}_{k-1}$, yields an approximation to the discrete information contents at the kth level from the discrete information contents at level $k-1$. Thus,

$$\mathcal{D}_k^{k-1} : V^k \to V^{k-1}, \qquad \mathcal{P}_{k-1}^k : V^{k-1} \to V^k,$$

where $\{V^k\}$ is a sequence of finite spaces. We define the prediction error as

$$e^k = f^k - \mathcal{P}_{k-1}^k f^{k-1}.$$

The operators $\mathcal{D}_k^{k-1}$ and $\mathcal{P}_{k-1}^k$ can be used to construct a multiresolution pyramid. If f^k is the input, then one stage of decomposition results in the decimated signal f^{k-1} and the prediction error e^k. Because e^k is at the same sample rate as f^k, (f^{k-1}, e^k) redundantly represents f^k. This single decomposition stage is iterated on the decimated signal for a multiresolution representation $(f^0, e^1, \ldots, e^L)$.

If the operators $\mathcal{D}_k^{k-1}$ and $\mathcal{P}_{k-1}^k$ satisfy the consistence property, i.e.,

$$\mathcal{D}_k^{k-1}\mathcal{P}_{k-1}^k = I_{V^k}, \tag{1}$$

then, it is possible to design a non-redundant multiresolution decomposition. Let G_k be a detail encoder such that $d^k = G_k e^k$ is at half the sample rate of e^k, and let $\tilde{G}_k$ be the corresponding decoder such that $\tilde{G}_k G_k e^k = e^k$. Then (f^{k-1}, d^k) is a non-redundant representation of f^k. This single stage is iterated on the decimated signal for a multiresolution representation $(f^0, d^1, \ldots, d^L)$.

A multiresolution scheme within Harten's framework is characterized by six operators: The fundamental discretization $\mathcal{D}_k$ and reconstruction $\mathcal{R}_k$, the decimation operator $\mathcal{D}_k^{k-1}$, the prediction operator $\mathcal{P}_{k-1}^k$, and the detail operators G_k and $\tilde{G}_k$. The main classical problem in the MR schemes is to *design* a good prediction operator that we will obtain using statistical Learning techniques.

2.2 Learning-Based Multiresolution (LMR)

In multiresolution analysis, we predict the results of the kth scale from the $(k-1)$th scale without using the information of the k level. In order to compress the data, it is important that the components forming d^k are zero (or close to zero). We reformulate the problem as a typical Learning problem: Knowing $\mathcal{D}_{k-1}f$ and $\mathcal{D}_k f$, with $f \in \mathcal{F}$, find an approximation to $\mathcal{D}_k f$ from $\mathcal{D}_{k-1}f$. The underlying idea is very easy; in the classical multiresolution the discrete data f^k contains the same information than f^{k-1} and d^k, which represent non-redundant information present in f^k and non-predictable from f^{k-1} by the prediction operator $\mathcal{P}_{k-1}^k$. In order to construct this operator we do not use the information of f^k. We need details to be zero (or close to zero); for this we use f^k to obtain our prediction operator $\mathcal{P}_{k-1}^k$ and we transfer part of the information contained in d^k to $\mathcal{P}_{k-1}^k$. This allows us to obtain smaller details d^k using this prediction operator. In this case we have $f^k \equiv \{f^{k-1}, d^k, \mathcal{P}_{k-1}^k\}$. In this section we explain the Learning-based multiresolution problem following Aràndiga et al. [2]. The aim is to learn to predict, i.e., to adapt our operator to the data in each problem.

In a Learning problem there are two principal components:

1. Some input vectors $\tilde{x} \in \mathbb{R}^n$.
   In Learning-based MR our random vectors are the data in the $(k-1)$th scale, $f^{k-1} = \mathcal{D}_k^{k-1}(f^k) \in V^{k-1}$. These vectors are our input vectors.
2. A supervisor (S) returns an output value y to every input vector $\tilde{x}$.
   In Learning-based MR our output value is each value in the kth scale, i.e., each value of $f^k \in V^k$.

The problem in LMR is the following: from the given class of continuous linear operators $(\mathcal{K}, ||\cdot||)$ find the one which best approximates the supervisor's response.

In Learning-based MR our prediction operator $\mathcal{P}_{k-1}^k$ is the minimum of the empirical risk functional, i.e., our general problem can be described as the minimization of the functional:

$$\mathcal{P}_{k-1}^k = \underset{G \in \mathcal{K}, ||G|| \leq M}{\arg\min} \; \mathcal{L}(f^k, G(f^{k-1})), \tag{2}$$

where M is a fixed parameter, $(\mathcal{K}, ||\cdot||)$ is the class of continuous linear operators and $\mathcal{L}(y, G(\tilde{x})) = \| y - G(\tilde{x}) \|$ is the loss-function.

3 Learning-Based Multiresolution Schemes for Point-Value Discretization

In this section we specify how to construct a prediction operator in the LMR context ([2]). For this, we solve the learning problem which can be divided in four parts:

i. First, we need the data. For this, in the MR context, we have to specify the discretization operator.
ii. We have chosen the class of the continuous linear operators $(\mathcal{K}, ||\cdot||)$ to minimize the risk operator.
iii. Then, we have to choose the *loss-function* for the risk functional. Depending on this function we can obtain different prediction operators, §3.3.
iv. Finally, we have to solve the problem of risk minimization.

3.1 Learning-Based Multiresolution Schemes for Point-Value Discretization on $[0, 1]$

In order to understand which is the Learning process in the Multiresolution scheme, we explain the multiresolution schemes for point-value (PV). This is a classical context employed in several examples (see, e.g., [4,15]).

Let us consider $\mathcal{F} = \mathcal{B}[0,1]$, the space of bounded functions on the closed unit interval, and let X^L a uniform partition of $[0,1]$.

$$X^L = \{x_i^L\}_{i=0}^{J_L}, \quad x_0^L = 0, \quad x_i^L = ih_L, \quad h_L = 1/J_L, \quad J_L = 2^L J_0, \qquad (3)$$

where J_0 is some integer. We define the lower resolution grids $X^k = \{x_i^k\}_{i=0}^{J_k}$, $k = L-1, \dots, 0$, by the dyadic coarsening

$$x_i^{k-1} = x_{2i}^k, \quad i = 0, \dots, J_{k-1} := J_k/2. \qquad (4)$$

and we define $f^k = \mathcal{D}_k f$ by

$$f_i^k = (\mathcal{D}_k f)_i = f(x_i^k), \quad f^k = \{f_i^k\}_{i=0}^{J_k}. \qquad (5)$$

From $x_i^{k-1} = x_{2i}^k$ the decimation can be defined as:

$$f_i^{k-1} = (\mathcal{D}_k^{k-1} f^k)_i = f_{2i}^k, \quad i = 0, \dots J_{k-1}. \qquad (6)$$

In Learning problem we also consider the next level f^k and use this information to obtain the prediction operator. The input vector is the stencil

$$\mathcal{S}^{r,s}(f_i^{k-1}) = (f_{i-r}^{k-1}, \dots, f_{i+s}^{k-1}) =: \tilde{f}_{i,r,s}^{k-1},$$

and for each stencil $\tilde{f}_{i,r,s}^{k-1}$ we have an output value in the k-level f_{2i-1}^k. With the inputs and outputs we assemble a training set of observations $(\tilde{f}_{i,r,s}^{k-1}, f_{2i-1}^k), i = r, \dots, J_{k-1} - s$.

We take the class of operators $\mathcal{K}_n = \{g_a : \mathbb{R}^n \to \mathbb{R} : \quad g_a(\tilde{x}) =< \tilde{x}, a >: a \in \mathbb{R}^n\} \subset \Pi_n^1(\mathbb{R})$ with $n = r + s + 1$, where $< \cdot, \cdot >$ is the scalar product. It is easy to prove that if $a \in \mathbb{R}^n$ then $||g_a|| = ||a||$. We fix a constant M and we select the loss function $\mathcal{Q}_{\ell^p} = |y - g_a(\tilde{x})|^p$ with $1 \le p < \infty$. Then we formulate the problem as follows:

$$\tau = \underset{g_a \in \mathcal{K}_n, ||g_a|| \le M}{\arg\min} \sum_i \mathcal{Q}_{\ell^p}(f_{2i-1}^k, g_a(\tilde{f}_{i,r,s}^{k-1})) = \sum_i |f_{2i-1}^k - g_a(\tilde{f}_{i,r,s}^{k-1})|^p. \qquad (7)$$

or if $p = \infty$,

$$\tau = \underset{g_a \in \mathcal{K}_n, ||g_a|| \le M}{\arg\min} \max_i |f_{2i-1}^k - g_a(\tilde{f}_{i,r,s}^{k-1})|. \qquad (8)$$

Then we define the prediction operator as

$$\begin{cases} (\mathcal{P}_{k-1}^k f^{k-1})_{2i} = f_i^{k-1}, \\ (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1} = \tau(\tilde{f}_{i,r,s}^{k-1}). \end{cases} \qquad (9)$$

And finally the error is defined as

$$d_i^k = e_{2i-1}^k = f_{2i-1}^k - (\mathcal{P}_{k-1}^k f^{k-1})_{2i-1} = f_{2i-1}^k - \tau(\tilde{f}_{i,r,s}^{k-1}).$$

3.2 LMR Methods versus Linear Piecewise Interpolation Methods

We review the classical interpolation methods in the MR context. By the consistence property (1) and by the definition of the discretization operator (5) a reconstruction procedure is given by any operator $\mathcal{R}_k$ such that $\mathcal{D}_k\mathcal{R}_k f^k = f^k$, i.e. $(\mathcal{R}_k f^k)(f_i^k) = f_i^k = f(x_i^k)$. Thus, $(\mathcal{R}_k f^k)(x)$ have to be a bounded function that interpolates the set $\{f_i^k\}_{i=0}^{J_k}$ at the nodes $\{x_i^k\}_{i=0}^{J_k}$.

Using piecewise centered polynomial interpolation we obtain that if

$$\mathcal{S}^{r,s}(x_i^{k-1}) = (x_{i-r}^{k-1}, \ldots, x_{i+s}^{k-1})$$

is the stencil of points used with $s = r - 1$ (centered) then

$$(\mathcal{P}_{k-1}^k f^{k-1})_{2i-1} = \sum_{l=1}^{r} \beta_l (f_{i+l-1}^{k-1} + f_{i-l}^{k-1}) \tag{10}$$

where the coefficients β_l are

$$\begin{cases} r = 1 \Rightarrow \beta_1 = \frac{1}{2}, \\ r = 2 \Rightarrow \beta_1 = \frac{9}{16}, \beta_2 = -\frac{1}{16}, \\ r = 3 \Rightarrow \beta_1 = \frac{150}{256}, \beta_2 = -\frac{25}{256}, \beta_3 = \frac{3}{256}. \end{cases}$$

In all the cases

$$(\mathcal{P}_{k-1}^k f^{k-1})_{2i} = f_i^{k-1}. \tag{11}$$

If we use interpolation techniques (PV) the filters are always the same and they are independent of the data that we have. On the other hand, with the LMR methods we adapt the filters to the data. In order to appreciate these differences we introduce the next example.

Example. Let the function

$$f(x) = (x + 1/4)^2 \sin(40\pi(x + 1/2)).$$

We discretize it in $J_k = 32$ values in the interval $[0, 1]$ with the operator $\mathcal{D}_5$ defined in (5). We show these data in Table 2. We choose a centered stencil with $s = 1$ and $r = 2$, so the class of functions will be $\mathcal{K}_4$. Therefore, for each input vector of the level $k - 1$ we have an output value of the level k (Table 1).

Remark 1. In the boundary we use piecewise cubic interpolation not centered for both methods: PV and LMR.

Table 1. Training set of observations: Input vectors and output values for the Learning process

Node i	Input Vector $\tilde{f}^{k-1}_{i,2,1}$	Output value f^k_{2i-1}
2	$(0, 0.0977, 0, -0.1914)$	-0.0836
3	$(0.0977, 0, -0.1914, 0)$	0.1167
4	$(0, -0.1914, 0, 0.3164)$	0.1554
5	$(-0.1914, 0, 0.3164, 0)$	-0.1996
6	$(0, 0.3164, 0, -0.4727)$	-0.2493
7	$(0.3164, 0, -0.4727, 0)$	0.3045
8	$(0, -0.4727, 0, 0.6602)$	0.3653
9	$(-0.4727, 0, 0.6602, 0)$	-0.4316
10	$(0, 0.6602, 0, -0.8789)$	-0.5034
11	$(0.6602, 0, -0.8789, 0)$	0.5807
12	$(0, -0.8789, 0, 1.1289)$	0.6636
13	$(-0.8789, 0, 1.1289, 0)$	-0.7520
14	$(0, 1.1289, 0, -1.4102)$	-0.8459
15	$(1.1289, 0, -1.4102, 0)$	0.9453

We choose as loss function $\mathcal{Q}_{\ell^2}$, i.e. we pose the typical least squares problem. Finally we take the bound for the norm of the prediction operator $M = 1$. Therefore, $a = (0.2021, -0.5556, -0.5087, 0.1552)^T$, where

$$a = \min_{\tilde{a}\in\mathbb{R}^4, ||\tilde{a}||_2\leq 1} \sum_{i=2}^{15} |f^k_{2i-1} - (a_1 f^{k-1}_{i-2} + a_2 f^{k-1}_{i-1} + a_3 f^{k-1}_{i} + a_4 f^{k-1}_{i+1})|^2$$

We define the LMR prediction operator for this example as:

$$\begin{cases} (\mathcal{P}^k_{k-1} f^{k-1})_{2i} = f^{k-1}_i, \\ (\mathcal{P}^k_{k-1} f^{k-1})_{2i-1} = 0.2021 f^{k-1}_{i-2} - 0.5556 f^{k-1}_{i-1} - 0.5087 f^{k-1}_{i} + 0.1552 f^{k-1}_{i+1}. \end{cases} \tag{12}$$

In the compress context when we use the LMR method we have to store the details and also the filters values. We will denote by FN (filters number) the number of the filters values to store. In this case FN= 4.

Now, from f^{k-1} (second column of the Table 2) we try to predict f^k (second column of the Table 2) using the polynomial interpolator ((10) with $r = 2$ and (11)) obtaining the sixth column of the Table 2 and using the LMR prediction operator (12) obtaining the third column. We observe that the details obtained with the LMR method are much smaller that those obtained with the PV method.

In the next section we explain the choice of the loss-function and with this we develop some theoretical properties.

Table 2. Point-value reconstruction using the LMR$_4$ and the cubic linear interpolation (PV$_4$) methods. There are some important differences in the error vector. The LMR method adapts the prediction operator to the output values.

		LMR$_4$			PV$_4$		
f^k	f^{k-1}	$\mathcal{P}_{k-1}^{k} f^{k-1}$	e^k	d^k	$\mathcal{P}_{k-1}^{k} f^{k-1}$	e^k	d^k
0	0	0	0		0	0	
−0.0559		0.0181	−0.0740	−0.0740	0.0181	−0.0740	−0.0740
0.0977	0.0977	0.0977	0		0.0977	0	
−0.0836		−0.0840	0.0004	0.0004	0.0669	−0.1504	−0.1504
0	0	0	0		0	0	
0.1167		0.1171	−0.0004	−0.0004	−0.1138	0.2305	0.2305
−0.1914	−0.1914	−0.1914	0		−0.1914	0	
0.1554		0.1555	−0.0001	−0.0001	−0.1274	0.2828	0.2828
0	0	−0.0000	0		0	0	
−0.1996		−0.1997	0.0001	0.0001	0.1899	−0.3895	−0.3895
0.3164	0.3164	0.3164	0		0.3164	0	
−0.2493		−0.2492	−0.0001	−0.0001	0.2075	−0.4568	−0.4568
0	0	0	0		0	0	
0.3045		0.3044	0.0001	0.0001	−0.2856	0.5902	0.5902
−0.4727	−0.4727	−0.4727	0		−0.4727	0	
0.3653		0.3651	0.0002	0.0002	−0.3071	0.6724	0.6724
0	0	−0.0000	0		−0.0000	0	
−0.4316		−0.4314	−0.0002	−0.0002	0.4009	−0.8325	−0.8325
0.6602	0.6602	0.6602	0		0.6602	0	
−0.5034		−0.5032	−0.0002	−0.0002	0.4263	−0.9297	−0.9297
0	0	0.0000	0		0	0	
0.5807		0.5806	0.0002	0.0002	−0.5356	1.1164	1.1164
−0.8789	−0.8789	−0.8789	0		−0.8789	0	
0.6636		0.6636	0.0000	0.0000	−0.5649	1.2285	1.2285
0	0	0	0		0	0	
−0.7520		−0.7519	−0.0000	−0.0000	0.6899	−1.4419	−1.4419
1.1289	1.1289	1.1289	0		1.1289	0	
−0.8459		−0.8461	0.0002	0.0002	0.7231	−1.5690	−1.5690
0	0	0	0		0	0	
0.9453		0.9456	−0.0002	−0.0002	−0.8638	1.8091	1.8091
−1.4102	−1.4102	−1.4102	0		−1.4102	0	
1.0503		−0.1587	1.2090	1.2090	−0.1587	1.2090	1.2090
0	0	0	0		0	0	

3.3 The Loss-Function in LMR Schemes

The choice of the loss function is very important for the LMR scheme. We can choose several possible functions because we work with the set of linear operators. The typical choice is the ℓ^p- norm, i.e., $\mathcal{Q}_{\ell^p}(y, g_a(\tilde{x})) = |y - g_a(\tilde{x})|^p$ with $1 \le p < \infty$; so, our problem is:

$$\tau(\tilde{x}) = \operatorname*{arg\,min}_{g_a \in \mathcal{K}, ||g_a|| \le M} \sum_i \mathcal{Q}_{\ell^p}(f^k_{2i-1}, g_a(\tilde{f}^{k-1}_{i,r,s})) = \operatorname*{arg\,min}_{g_a \in \mathcal{K}, ||g_a|| \le M} \sum_i |f^k_{2i-1} - g_a(\tilde{f}^{k-1}_{i,r,s})|^p, \tag{13}$$

where $\mathcal{K}_n = \{g_a : \mathbb{R}^n \to \mathbb{R} : \quad g_a(\tilde{x}) = \sum_{j=1}^n a_j x_j \, : \, a_j \in \mathbb{R}, \; 1 \le j \le n\}$ with $n = r + s + 1$. Then:

$$a = \min_{\tilde{a} \in \mathbb{R}^n, ||\tilde{a}|| \le M} \sum_i |f^k_{2i-1} - < \tilde{f}^{k-1}_{i,r,s}, \tilde{a} > |^p = \min_{\tilde{a} \in \mathbb{R}^n, ||\tilde{a}|| \le M} ||f^k - \mathbf{F}^{k-1}_{r,s} \tilde{a}||^p_p, \tag{14}$$

where $\tilde{f}^k = (f^k_1, f^k_3, \ldots, f^k_{J_k - 1})^T$ and $\mathbf{F}^{k-1}_{r,s}$ is a matrix with the i-row equal to $\tilde{f}^{k-1}_{i,r,s}$.

If $p = \infty$ then the problem (14) is:

$$\tilde{b} = \min_{\tilde{a}\in\mathbb{R}^n, ||\tilde{a}||\leq M} ||f^k - \mathbf{F}^{k-1}_{r,s}\tilde{a}||_\infty = \min_{\tilde{a}\in\mathbb{R}^n, ||\tilde{a}||\leq M} \max_i |f^k_{2i-1} - < \tilde{f}^{k-1}_{i,r,s}, \tilde{a} > |, \quad (15)$$

that is the classical Chebyshev problem.

When $p = 2$ is the typical least squares problem. We can reformulate the ℓ^p problem as a second-order cone programs (SOCP) (see, e. g., [5]) and if the dimensions of the system are not very large we can solve it using the `cvx` package designed by Grant et al. [12,13]. In this work we will only use $p = 1, 2, \infty$.

3.4 Some Properties of LMR Context

In this section we study some properties of LMR. For this, we present two important definitions.

Definition 1. *Let the set of polynomials of degree less than or equal to* r

$$p \in \Pi^r_1(\mathbb{R}) = \{g \,|\, g_a(x) = \sum_{i=0}^{r} a_i x^i,\ a_i \in \mathbb{R},\ \forall\, i\}.$$

Then the order of the prediction operator is $r + 1$ *if*

$$\mathcal{P}^k_{k-1}(\mathcal{D}_{k-1}p) = \mathcal{D}_k p, \quad (16)$$

i.e., the prediction operator is exact for polynomials of degree less than or equal to r.

Theorem 1. *The order of the prediction operator* $\mathcal{P}^k_{k-1}$ *obtained by LMR schemes with the class of functions* $\mathcal{K}_n$ *and the loss function* $\mathcal{Q}_{\ell^p}$, $1 \leq p \leq +\infty$ *and* $n \leq M$ *is at least* $n + 1$.

Proof. We are going to obtain a solution for the problems (14) and (15) that belongs to $\mathcal{K}_n$ and it is of order $n + 1$. Therefore, let the set of points

$$\mathcal{S}^{r,s}(x^{k-1}_i) = \{x^{k-1}_{i-r}, \ldots, x^{k-1}_{i+s}\},$$

and $\{L_m(x)\}_{m=-r,\ldots,s}$ is the Lagrange interpolation polynomials of grid points $\mathcal{S}^{r,s}(x^{k-1}_i)$. Then, we define

$$q^{k-1}_i(x; f^{k-1}, r, s) := \sum_{m=-r}^{s} f^{k-1}_{i+m} L_m\left(\frac{x - x^{k-1}_{i+m}}{h_{k-1}}\right).$$

Thus

$$\mathcal{I}_{k-1}(x; f^{k-1}) = q^{k-1}_i(x) \quad x \in [x^{k-1}_{i-1}, x^{k-1}_i],\ 1 \leq i \leq J_{k-1}$$

and we define the prediction operator as:

$$(\mathcal{P}^k_{k-1} f^{k-1})_i = \mathcal{I}_{k-1}(x^k_i; f^{k-1}),$$

a straightforward algebraic manipulation shows that

$$\begin{array}{l}(\mathcal{P}_{k-1}^{k} f^{k-1})_{2i} = f_i^{k-1}, \\ (\mathcal{P}_{k-1}^{k} f^{k-1})_{2i-1} = \sum_{m=-r}^{s} L_m(-1/2) f_{i+m}^{k-1}.\end{array} \tag{17}$$

This prediction operator is of order $n+1$, with $n = r+s+1$. Then let

$$\tilde{b} = (L_{-r}(-1/2), \ldots, L_s(-1/2)),$$

for some polynomial p of degree n we have that

$$|p_{2i-1}^{k} - < p_{i,r,s}^{k-1}, \tilde{b} > | = 0, \quad \forall\, 1 \leq i \leq J_{k-1}, \forall\, k \tag{18}$$

and as $|L_m(-1/2)| \leq 1$, $\forall\, m = -r, \ldots, s$ then $||\tilde{b}||_p \leq n$ with $p = 1, \ldots, \infty$. Therefore, $\tilde{b}$ is a solution of the problems (14) and (15). □

Definition 2 (Stability of the multiresolution). *The decomposition algorithm is stable with respect to the norm* $||\cdot||$ *if* $\exists\ C$ *such that* $\forall\ k$, $\forall\ f^k \mapsto (f^{J_0}, d^1, \ldots, d^k)$ *and* $\hat{f}^k \mapsto (\hat{f}^{J_0}, \hat{d}^1, \ldots, \hat{d}^k)$*, then*

$$\begin{aligned} ||f^{J_0} - \hat{f}^{J_0}|| &\leq C||f^k - \hat{f}^k|| \\ ||d^m - \hat{d}^m|| &\leq C||f^k - \hat{f}^k||, \quad \forall\, 1 \leq m \leq k. \end{aligned}$$

The reconstruction algorithm is stable with respect to the norm $||\cdot||$ *if* $\exists\, C$ *such that* $\forall\ k > J_0$, $\forall\ (f^{J_0}, d^1, \ldots, d^k) \mapsto f^k, (\hat{f}^{J_0}, \hat{d}^1, \ldots, \hat{d}^k) \mapsto \hat{f}^k$*:*

$$||f^k - \hat{f}^k|| \leq C \sup(||f^{k-1} - \hat{f}^{k-1}||, ||d^k - \hat{d}^k||). \tag{19}$$

The question of stability is, of course, crucial when one intends to use the MR transform in a signal compression context. Indeed, the difference between the initial signal and the reconstructed signal after compression is controlled by stability constants.

Theorem 2. *The reconstruction algorithm with a bounded linear prediction operator is stable.*

Proof. First, we have that

$$||f^k - \hat{f}^k|| = ||\mathcal{P}_{k-1}^{k} f^{k-1} - \mathcal{P}_{k-1}^{k} \hat{f}^{k-1} + d^k - \hat{d}^k|| \leq ||\mathcal{P}_{k-1}^{k}|| ||f^{k-1} - \hat{f}^{k-1}|| + ||d^k - \hat{d}^k||, \tag{20}$$

as the prediction operator is bounded, we have a constant $M > 0$ such that $||\mathcal{P}_{k-1}^{k}|| \leq M$. Therefore,

$$||f^k - \hat{f}^k|| \leq (M+1) \sup(||f^{k-1} - \hat{f}^{k-1}||, ||d^k - \hat{d}^k||). \quad \square \tag{21}$$

Corollary 1. *The reconstruction algorithm with the prediction operator* $\mathcal{P}_{k-1}^{k}$ *obtained by LMR schemes with the class of functions* $\mathcal{K}_n$ *and the loss function* $\mathcal{Q}_{\ell^p}$, $1 \leq p \leq +\infty$ *(problems (14) and (15)) in the PV context is stable.*

4 Numerical Experiments

We consider multiresolution schemes in the point-value framework using reconstruction operators obtained from Lagrange polynomial interpolatory techniques; using LMR techniques using loss function $\mathcal{Q}_{\ell^p}$ with $p = 1, 2, \infty$ and the PPH reconstruction.

Each multiresolution scheme is identified by an acronym. The equivalences are as follows:

PV$_4$: Point-value with interpolatory techniques using four centered points. The prediction operator is:

$$\begin{cases} (\mathcal{P}_{k-1}^{k} f^{k-1})_{2i} = f_i^{k-1}, \\ (\mathcal{P}_{k-1}^{k} f^{k-1})_{2i-1} = -\frac{1}{16} f_{i-2}^{k-1} + \frac{9}{16} f_{i-1}^{k-1} + \frac{9}{16} f_i^{k-1} - \frac{1}{16} f_{i+1}^{k-1}. \end{cases}$$

PPH: This scheme introduced by Amat et al. [1] is a nonlinear stable algorithm which consists in modifying the PV$_4$ scheme:

$$\begin{cases} (\mathcal{P}_{k-1}^{k} f^{k-1})_{2i} = f_i^{k-1}, \\ (\mathcal{P}_{k-1}^{k} f^{k-1})_{2i-1} = \frac{1}{2}(f_{i-1}^{k-1} + f_i^{k-1}) - \frac{1}{8} pph(d^2 f_i^{k-1}, d^2 f_{i-1}^{k-1}). \end{cases}$$

where $d^2 f_i^{k-1} = f_{i+1}^{k-1} - 2 f_i^{k-1} + f_{i-1}^{k-1}$ and pph is the function

$$pph(x, y) = \left(\frac{sign(x) + sign(y)}{2} \right) \frac{2|x||y|}{|x| + |y|}, \quad \forall\, x, y \in \mathbb{R} \setminus \{0\};$$
$$pph(x, 0) = 0,\ \forall\, x \in \mathbb{R}; \qquad pph(0, y) = 0,\ \forall\, y \in \mathbb{R}.$$

^{p}LMR$_m$: LMR methods with the ℓ^p norm chosen for the loss function ($p = 1, 2, \infty$) and with m points used (here we will use $m = 4$ centered points).

We take $J_0 = 17$ and $J_L = 1025$, thus $L = 6$. In the LMR case we have to store the filters of each level as we have indicated in §3.1. In §3.2 we have denoted as FN (filters number). In this paper we take $m = 4$ and $L = 6$ then FN$= L \times m = 24$.

We truncate our details as follows:

$$\hat{d}_i^k = \begin{cases} d_i^k, \text{ if } |d_i^k| \geq \varepsilon; \\ 0, \ \text{ if } |d_i^k| < \varepsilon. \end{cases}$$

with $i = 0, \ldots, J_k$, where ϵ is the threshold parameter introduced. We measure the error in the ℓ^p discrete norm with $p = 1, 2, \infty$, so

$$E_1 = \frac{1}{J_L} \sum_i |f_i^L - \hat{f}_i^L|, \quad E_2 = \sqrt{\frac{1}{J_L} \sum_i |f_i^L - \hat{f}_i^L|^2}, \quad E_\infty = ||f^L - \hat{f}^L||_\infty.$$

A measure of the compression capabilities of the scheme can be given by the factor:

$$r_c = \frac{J_L - J_0 - (\text{NNZ} + \text{FN})}{J_L - J_0} \tag{22}$$

where NNZ is the number of details different to zero ($\mathcal{D}_\varepsilon = \{(i,k) : |d_i^k| > \varepsilon_k\}$). In the LMR case we also include the number of filters, FN.

We attain full compression ($r_c = 1$) when NNZ=0 and FN=0. When NNZ=$J_L - J_0$ (and FN=0) we have no compression ($r_c = 0$).

The discrete data to be analyzed consist of the point-values of different functions, f_1, f_2, f_3, on the finest grid (see Fig. 1). We use different types of functions: oscillatory, oscillatory with a smooth part and oscillatory with a smooth part and a discontinuity. Compressing this type of functions tends to be difficult using interpolation techniques because it needs more points to correctly interpolate each wave.

$$f_1(x) = 40(x+1/4)^2 \sin(40\pi(x+1/2)),$$

$$f_2(x) = \begin{cases} 40\sin(40\pi x), & 0 \le x \le 7/20; \\ -1.96, & 7/20 \le x \le 267/500; \\ 10\sin(30\pi x) - 2, & 267/500 \le x \le 1, \end{cases}$$

$$f_3(x) = \begin{cases} \sin(0.2x), & x \in [-2\pi, 0[\cup [2\pi, 4\pi[; \\ \sin(0.2x) + 0.2\sin(10x), & x \in [0, 2\pi[; \\ \sin(0.2x) + 1, & \text{in other case.} \end{cases}$$

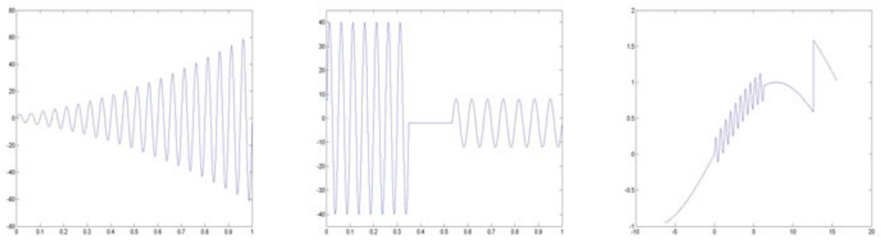

(a) (b) (c)

Fig. 1. Functions: (a) f_1, (b) f_2, (c) f_3

The function f_1 is specially difficult to compress with linear methods (PV) as we can see in Table 3. In the LMR methods the prediction operator has much information; thus, we can see that these methods have a high compression rate because the part of information is transfered from the details to the prediction operator. There are not differences between the methods with $\mathcal{Q}_{\ell^1}$ and $\mathcal{Q}_{\ell^2}$.

In the next example, function f_2, we can see that if the function has an oscillatory zone and a smooth zone, the loss function $\mathcal{Q}_{\ell^2}$ acts only about the oscillatory zone and it losses information about the smooth zone, so this scheme does not offer good results.

We have chosen the function f_1 because the training set is similar. In f_2 we introduce a constant part with two oscillatory parts. In this example, we will see that the LMR scheme with loss function $\mathcal{Q}_{\ell^2}$ needs many details to obtain a

Table 3. Function f_1. Errors and number of details obtained with various compression schemes.

	$m = 4, \quad \varepsilon = 10^{-3}$					
	E_1	E_2	E_∞	FN	NNZ	r_c
PV_4	$1.16 \cdot 10^{-4}$	$2.66 \cdot 10^{-4}$	$9.97 \cdot 10^{-4}$	0	741	0.264
${}^\infty\text{LMR}_4$	$1.20 \cdot 10^{-4}$	$2.12 \cdot 10^{-4}$	$1.17 \cdot 10^{-3}$	24	95	0.882
${}^2\text{LMR}_4$	$1.83 \cdot 10^{-4}$	$2.95 \cdot 10^{-4}$	$9.97 \cdot 10^{-4}$	24	79	0.897
${}^1\text{LMR}_4$	$1.80 \cdot 10^{-4}$	$2.80 \cdot 10^{-4}$	$9.33 \cdot 10^{-4}$	24	66	0.910
PPH	$4.87 \cdot 10^{-5}$	$1.81 \cdot 10^{-4}$	$9.99 \cdot 10^{-4}$	0	925	0.082

Table 4. Function f_2. Errors and number of details obtained with various compression schemes.

	$m = 4, \quad \varepsilon = 10^{-3}$					
	E_1	E_2	E_∞	FN	NNZ	r_c
PV_4	$7.36 \cdot 10^{-5}$	$1.72 \cdot 10^{-4}$	$9.83 \cdot 10^{-4}$	0	554	0.450
${}^\infty\text{LMR}_4$	$3.48 \cdot 10^{-6}$	$4.82 \cdot 10^{-5}$	$9.01 \cdot 10^{-4}$	24	1002	−0.017
${}^2\text{LMR}_4$	$8.71 \cdot 10^{-6}$	$7.40 \cdot 10^{-5}$	$8.56 \cdot 10^{-4}$	24	991	−0.006
${}^1\text{LMR}_4$	$2.69 \cdot 10^{-4}$	$4.43 \cdot 10^{-4}$	$9.67 \cdot 10^{-4}$	24	**375**	0.604
PPH	$8.09 \cdot 10^{-5}$	$2.05 \cdot 10^{-4}$	$9.79 \cdot 10^{-4}$	0	639	0.366

Table 5. Function f_3. Errors and number of details obtained with various compression schemes.

	$m = 4, \quad \varepsilon = 10^{-3}$					
	E_1	E_2	E_∞	FN	NNZ	r_c
PV_4	$5.37 \cdot 10^{-5}$	$1.24 \cdot 10^{-4}$	$9.30 \cdot 10^{-4}$	0	165	0.836
${}^\infty\text{LMR}_4$	$1.20 \cdot 10^{-4}$	$2.88 \cdot 10^{-4}$	$1.55 \cdot 10^{-3}$	24	832	0.150
${}^2\text{LMR}_4$	$1.97 \cdot 10^{-4}$	$3.71 \cdot 10^{-4}$	$1.55 \cdot 10^{-3}$	24	670	0.311
${}^1\text{LMR}_4$	$6.72 \cdot 10^{-5}$	$1.20 \cdot 10^{-4}$	$5.47 \cdot 10^{-4}$	24	**89**	0.887
PPH	$1.16 \cdot 10^{-5}$	$2.24 \cdot 10^{-4}$	$1.58 \cdot 10^{-3}$	0	164	0.837

good approximation. We have more quantity of NNZ using LMR methods with $p = 2, \infty$. However, if we use the loss function $\mathcal{Q}_{\ell^1}$ we obtain the third part of the details than the PV method with lower errors E_2, E_1, E_∞. We can see this in Table 4.

In the last experiment, function f_3 (Table 5), we introduce an oscillatory part, a smooth part and a discontinuity. In Table 5, we can see that the results obtained with ${}^1\text{LMR}_4$ are really good, taking into account that the signal is quite difficult to compress. The other methods obtain worse results than the linear method (PV) and PPH method.

In conclusion the prediction operator using the loss function $\mathcal{Q}_{\ell^2}$ reduces the errors in the difficult zones of the function as the oscillatory zones. Problems can arise when the function presents oscillatory zones and smooth zones; then, the method ^{2}LMR losses its efficiency. In order to solve this problem, we introduce the robust norm $\mathcal{Q}_{\ell^1}$ and we obtain a highly compression rate in every type of signals.

5 Conclusions and Future Research

In this work, we use the method developed by Aràndiga et al. (see, e.g., [2,18]) based on Harten's framework [15]. It consists in adapting our prediction operator to the data of the level that we have to predict, i.e., knowing the result find the best prediction operator in an admissible class, $\mathcal{K}$. In order to obtain our prediction operator we use a loss function, $\mathcal{Q}$, to measure the difference between the real value and our approximate value. So, the LMR problem is to find an operator within the class $\mathcal{K}$ which minimizes the loss function $\mathcal{Q}$.

Therefore, we integrate ([2]) one new idea from a new field, Learning statistical theory, into Harten's framework.

We use this method to compress signals with some conclusions:

i. The method obtains a high compression rate in signals that are difficult to compress.
ii. We adapt our prediction operator to the data. Therefore, we have to store the coefficients of this operator (FN).
iii. When the training set has similar characteristics, then the method with some ℓ^p-norm produces good results.
iv. When the training set has not similar characteristics, then the only method which produces great results is when the loss function is the ℓ^1 norm.

References

1. Amat, S., Donat, R., Liandrat, J., Trillo, J.C.: A fully adaptive PPH multiresolution scheme for image processing. Math. Comput. Modelling 46, 2–11 (2007)
2. Aràndiga, F., Cohen, A., Yáñez, D.F.: Learning-based multiresolution: Application to compress digital images (in preparation)
3. Aràndiga, F., Donat, R., Harten, A.: Multiresolution based on weighted averages of the hat function II: Non-linear reconstruction technique. SIAM J. Numer. Anal. 20, 1053–1093 (1999)
4. Aràndiga, F., Donat, R.: Nonlinear multiscale decompositions: The approach of A. Harten. Numerical Algorithms 23, 175–216 (2000)
5. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, New York (2004)
6. Cohen, A., Dahmen, W., Daubechies, I., DeVore, R.: Tree approximation and optimal encoding. Appl. Comp. Harm. Anal. 11, 192–226 (2001)
7. Cohen, A., Daubechies, I., Feauveau, J.: Biorthogonal bases of compactly supported wavelets. Comm. Pure Appl. Math. 45, 485–560 (1992)
8. Cohen, A.: Numerical Analysis of Wavelet Methods. Elsevier, Paris (2003)
9. Donoho, D.L.: Unconditional bases are optimal bases for data compression and for statistical estimation. Appl. Comp. Harm. Annal. 1, 110–115 (1993)

10. Friedman, J., Hastie, T., Tibshirani, R.: The Elements of Statistical Learning. Springer, New York (2001)
11. Getreuer, P., Meyer, F.: ENO multiresolution schemes with general discretizations. SIAM J. Numer. Anal. 46, 2953–2977 (2008)
12. Grant, M., Boyd, S.: Graph implementations for nonsmooth convex programs. In: Blondel, V., Boyd, S., Kimura, H. (eds.) Recent Advances in Learning and Control (a tribute to M. Vidyasagar). LNCIS, vol. 371, pp. 95–110. Springer, Heidelberg (2008)
13. Grant, M., Boyd, S.: CVX: Matlab software for disciplined convex programming (web page and software) (2009), http://stanford.edu/~boyd/cvx
14. Harten, A.: Discrete multiresolution analysis and generalized wavelets. J. Appl. Numer. Math. 12, 153–192 (1993)
15. Harten, A.: Multiresolution Representation of data: General framework. SIAM J. Numer. Anal. 33, 1205–1256 (1996)
16. Mallat, S.: A wavelet tour of signal processing. Academic Press, London (1998)
17. Vapnik, V.N.: The Nature of Statistical Learning. Springer, New York (1995)
18. Yáñez, D.F.: Learning Multiresolution: Transformaciones Multiescala derivadas de la Teoría estadística de Aprendizaje, Phd. Thesis, University of Valencia (2010)

Weighted-Power$_p$ Nonlinear Subdivision Schemes

Francesc Aràndiga, Rosa Donat, and Maria Santágueda

Depto. Matematica Aplicada. Universitat de Valencia
Avda Dr Moliner 50, 46100 Burjassot, Spain
{arandiga,donat,m.santagueda}@uv.es
http://gata.uv.es/

Abstract. In this paper we present and analyze a generalization of the *Power$_p$* subdivision schemes proposed in [3,12]. The *Weighted-Power$_p$* schemes are based on a harmonic weighted version of the *Power$_p$* average considered in [12], and their development is motivated by the desire to generalize the nonlinear analysis in [3,5] to interpolatory subdivision schemes with higher than second order accuracy.

Keywords: Nonlinear subdivision scheme, convergence, stability, regularity.

1 Introduction

Subdivision schemes are a powerful tool for the fast generation of curves and surfaces in computer-aided geometric design, as well as an essential ingredient in many multiscale algorithms used in data compression. The convergence properties of the recursive process specified by a subdivision scheme is of great importance in image, or surface, generation, as well as in image compression, hence it has been, and continues being, the subject of active research.

Many classical, data-independent, subdivision schemes can be described as linear operators between spaces of sequences. The convergence properties of linear subdivision schemes is a well understood subject nowadays (see e.g. [9,15] and references therein). The inability of linear subdivision schemes to reconstruct 'discontinuous' discrete data without creating spurious oscillations is also a well known deficiency which motivates the search for nonlinear schemes with specific properties.

In recent years, several nonlinear subdivision schemes have been proposed (see e.g.[4] and references therein) in an attempt to avoid the Gibb's-like oscillatory behavior that occurs when linear techniques are used to refine discrete data that displays a nearly discontinuous behavior. The analysis of nonlinear subdivision schemes cannot be carried out with the same techniques as in the linear case. A successful way to study the convergence properties of a nonlinear subdivision scheme follows from the ability to write the nonlinear scheme as a perturbation of a standard linear subdivision scheme whose convergence properties are known. It is shown in [5] that, in such cases, convergence can be obtained if the perturbation satisfies certain contraction properties.

In this paper we propose a new class of nonlinear subdivision schemes that generalize the *Power*$_p$ interpolatory subdivision schemes analyzed in [3,12]. Our schemes are based on a weighted harmonic mean, instead of the regular harmonic mean used in [3,12], in the definition of the nonlinear part of the scheme. Our motivation for considering the weighted harmonic mean stems from the desire to construct nonlinear versions of the six-point Deslauries-Dubuc interpolatory schemes, in a similar way to the construction of the *PPH* scheme in [4,5,6], or the *Power*$_p$ schemes, as non-linear counterparts to the 4-point Deslauries-Dubuc interpolatory scheme.

The paper is organized as follows: in Section 2 we recall several well known linear and nonlinear subdivision schemes, in order to motivate the introduction of the nonlinear weighted mean studied in this paper. We also recall the main results about interpolatory subdivision schemes which are used to study the proposed nonlinear schemes. In Section 3 we briefly recall the schemes *Power*$_p$ and in Section 4 we introduce the *Weighted-Power*$_p$ mean, and the corresponding subdivision schemes. We also study their properties with respect to convergence, stability and polynomial reproduction. Finally, in Section 5 we present a numerical example that shows that the proposed subdivision schemes also behave in a non-oscillatory manner when refining discrete discontinuous data. We close in Section 6 with some conclusions and future perspectives.

2 Linear and Nonlinear Interpolatory Subdivision Schemes

In subdivision algorithms, discrete data are recursively generated from coarse to fine scales by means of local rules. A linear (uniform) subdivision scheme S is defined as follows

$$(Sf)_n = \sum_{m\in\mathbb{Z}} a_{n-2m} f_m, \quad \forall f \in l^\infty(\mathbb{Z}), \forall n \in \mathbb{Z}.$$

The mask of the scheme, i.e. the sequence $\{a_n\}$, is of finite length, and characterizes the subdivision procedure. Interpolatory subdivision schemes have the property that the set of control points at each level of refinement remain unchanged after the application of the refinement rules which, in terms of the mask means that $a_{2n} = \delta_{0,n}$.

An important class of linear (that is, data-independent) interpolatory subdivision schemes are the Deslauries-Dubuc (DD henceforth) subdivision schemes, [14]. As noticed by A. Harten [16], the local refinement rules of DD subdivision schemes can be obtained by evaluating a Lagrange interpolatory polynomial whose interpolation stencil is symmetric with respect to the evaluation point. As in [16], we shall refer to the Deslauries-Dubuc subdivision schemes whose interpolation stencil comprises l points on the left and l points on the right to the evaluation point as $S_{l,l}$. Most relevant to the contents of this paper are the two-point, $S_{1,1}$, the four point, $S_{2,2}$, and the six-point, $S_{3,3}$, DD interpolatory subdivision schemes, based on first, third and fifth order interpolatory polynomials. These schemes can be expressed as follows (see e.g. [16])

$$\begin{cases} (S_{1,1}f)_{2n} = f_n, \\ (S_{1,1}f)_{2n+1} = \frac{1}{2}f_n + \frac{1}{2}f_{n+1}. \end{cases} \tag{1}$$

$$\begin{cases} (S_{2,2}f)_{2n} = f_n, \\ (S_{2,2}f)_{2n+1} = -\frac{1}{16}f_{n-1} + \frac{9}{16}f_n + \frac{9}{16}f_{n+1} - \frac{1}{16}f_{n+2}. \end{cases} \tag{2}$$

$$\begin{cases} (S_{3,3}f)_{2n} = f_n, \\ (S_{3,3}f)_{2n+1} = \frac{3}{256}f_{n-2} - \frac{25}{256}f_{n-1} + \frac{150}{256}f_n + \frac{150}{256}f_{n+1} - \frac{25}{256}f_{n+2} + \frac{3}{256}f_{n+3}. \end{cases} \tag{3}$$

This paper is concerned with the study of a generalization of the nonlinear operator used in the construction of the nonlinear versions of the $S_{2,2}$ scheme defined in [4,3,12].

The $S_{2,2}$ scheme can also be written as

$$\begin{cases} (S_{2,2}f)_{2n} = f_n, \\ (S_{2,2}f)_{2n+1} = \frac{f_n+f_{n+1}}{2} - \frac{1}{8}\ \mathrm{mean}(\nabla^2 f_{n+1}, \nabla^2 f_n), \end{cases}$$

with $\mathrm{mean}(x,y)=\frac{x+y}{2}$ and $\nabla^2 f_k = f_{k+1}-2f_k+f_{k-1}$. The *PPH*[1] scheme proposed in [4] substitutes the arithmetic mean above by the *PPH*-mean of the same two values, where

$$PPH(x,y) = \frac{sign(x)+sign(y)}{2}\frac{2|x||y|}{|x|+|y|}. \tag{4}$$

The *Power$_p$* schemes are a generalization of the *PPH* scheme (which is seen to be a *Power$_p$* scheme, with $p = 2$). As shown in [4,12], these schemes manage to avoid Gibb's like behavior close to a discontinuity in the data because the *Power$_p$* mean stays close to the minimum of the two values if their relative size is very large. We can explain this behavior by rewriting the $S_{2,2}$ scheme as

$$S_{2,2} = \frac{1}{2}S_{2,1} + \frac{1}{2}S_{1,2},$$

where $S_{l,r}$ is the interpolatory subdivision scheme whose interpolation stencil spans l points to the left and r points to the right of the evaluation point. It can readily be seen that

$$(S_{1,2}f)_{2n+1} = \frac{1}{2}(f_n + f_{n+1}) - \frac{1}{8}\nabla^2 f_{n+1},$$
$$(S_{2,1}f)_{2n+1} = \frac{1}{2}(f_n + f_{n+1}) - \frac{1}{8}\nabla^2 f_n,$$

so that these nonlinear schemes stay closer to the $S_{2,1}$, or the $S_{1,2}$ scheme when there is a large gradient, thus avoiding the oscillations associated to interpolatory stencils that cross a discontinuity in the data.

[1] *PPH* stands for Piecewise Polynomial Harmonic, see [4].

In [8], we show that a similar reasoning can be carried out in order to design a non-linear version of the $S_{3,3}$ DD-scheme. In this case it is necessary to consider non-linear versions of weighted averages, where the weights are different from 1/2, which motivates the introduction of the *Weighted-Power*$_p$ mean defined in this paper. It is easy to see (see [12]) that $Power_2(x, y) = PPH(x, y)$. We shall define the weighted nonlinear mean in such a way that

$$WP_{p,\frac{1}{2},\frac{1}{2}}(x, y) = Power_p(x, y),$$

and study, in this paper, what properties of the $Power_p$ mean can also be extended to the weighted case, and under what conditions.

We shall consider next the simpler case of constructing a nonlinear version of the $S_{3,2}$ scheme, which is an interpolatory subdivision scheme that considers an interpolatory stencil composed of 3 points to the left and two points to the right of the interpolation point. It is not difficult to see that

$$S_{3,2} = \frac{3}{8}S_{3,1} + \frac{5}{8}S_{2,2},$$

and also that

$$(S_{3,1}f)_{2n+1} = \frac{1}{2}(f_n + f_{n+1}) - \left(\frac{3}{16}\nabla^2 f_n - \frac{1}{16}\nabla^2 f_{n-1}\right),$$

hence

$$(S_{3,2}f)_{2n+1} = \frac{1}{2}(f_n + f_{n+1}) - \frac{3}{8}A - \frac{5}{8}B, \tag{5}$$

with

$$A = \frac{1}{16}\left(3\nabla^2 f_n - \nabla^2 f_{n-1}\right), \quad B = \frac{1}{16}\left(\nabla^2 f_n + \nabla^2 f_{n+1}\right).$$

Hence, if we try to define a nonlinear version of the $S_{3,2}$ scheme that would remain as close as possible to the $S_{3,1}$ scheme, for example because we would like to diminish the influence of the singularity-crossing stencil, it would be more appropriate to consider weighted versions of the *PPH* mean, or Power$_p$ mean. The study performed in this paper can be considered as a first step in order to design and study non-linear versions of the $S_{3,2}$, and the $S_{3,3}$ schemes.

2.1 Convergence and Stability of a Subdivision Scheme

It is important to remark that the tools to analyze the convergence and stability properties of a nonlinear subdivision scheme are very different from those applied in the linear case. In this paper, we follow the framework in [4] in order to analyze the convergence properties of our proposed schemes. We give below the relevant definitions for the sake of completeness, see e.g. [2,3,4,12].

Definition 1. Convergence. *A subdivision operator is called uniformly convergent if, for every set of initial data $f^0 \in l_\infty(\mathbb{Z})$, there exists a continuous function $S^\infty f^0 \in C(\mathbb{R})$, such that*

$$\lim_{k\to\infty} ||S^k f^0 - S^\infty f^0(2^{-k}\cdot)||_{l_\infty(\mathbb{Z})} = 0.$$

Definition 2. $C^{\alpha-}$ **Convergence.** *A convergent subdivision scheme S is said to be $C^{\alpha-}$ convergent if $S^\infty f \in C^{\alpha-}$, for all $f \in l^\infty(\mathbb{Z})$. We recall (see e.g. [12]) that the space $C^{\alpha-}$, $0 < \alpha \leq 1$, is the space of bounded continuous functions such that there exists a constant $C > 0$ so that*

$$\forall x, y \in \mathbb{R}, \quad |f(x) - f(y)| \leq C|x-y|^{\alpha_1}, \quad \forall \alpha_1 < \alpha.$$

If $\alpha > 1$, $\alpha = p + r > 0$ with $p \in \mathbb{N}$ and $0 < r < 1$,

$$C^{\alpha-} = \{f : \quad f^{(p)} \in C^{r-}\},$$

where $f^{(p)}$ stands for the derivative of order p.

Definition 3. Stability. *A convergent subdivision operator is called stable, if there exists a constant C such that, for every pair of initial data f^0, $\widetilde{f}^0 \in l_\infty(\mathbb{Z})$,*

$$||S^\infty f^0 - S^\infty \widetilde{f}^0||_{L^\infty} \leq C||f^0 - \widetilde{f}^0||_{l_\infty(\mathbb{Z})}.$$

In [3], the authors develop a tool to analyze the convergence and stability of nonlinear subdivision schemes $S_{NL} : l^\infty(\mathbb{Z}) \to l^\infty(\mathbb{Z})$ that can be written as a nonlinear perturbation of a convergent linear scheme, i.e.,

$$\forall f \in l^\infty(\mathbb{Z}), \quad \begin{cases} (S_{NL})_{2n} = f_n, \\ (S_{NL})_{2n+1} = (Sf)_{2n+1} + F(\delta f)_{2n+1}, \end{cases} \quad \forall n \in \mathbb{Z}, \tag{6}$$

where F is a nonlinear operator defined on $l^\infty(\mathbb{Z})$, δ is a linear and continuous operator on $l^\infty(\mathbb{Z})$ and S is a linear subdivision scheme. The following results can be found in [3].

Theorem 1. *Let S_{NL} be a nonlinear subdivision scheme which can be written in the form (6), with S a convergent linear subdivision scheme. If F and δ satisfy the following conditions:*

$$\exists M > 0, \quad : \qquad \forall d \in l^\infty(\mathbb{Z}) \quad ||F(d)||_\infty \leq M||d||_\infty, \tag{7}$$

$$\exists L > 0, \exists T < 1 : \qquad \forall f \in l^\infty(\mathbb{Z}) \quad ||\delta S^L_{NL}(f)||_\infty \leq T||\delta f||_\infty, \tag{8}$$

then the subdivision scheme S_{NL} in (6) is uniformly convergent. Moreover, if S is $C^{\alpha-}$ convergent, then S_{NL} is $C^{\beta-}$ convergent with $\beta = \min\{-\frac{\log_2(T)}{L}, \alpha\}$.

Theorem 2. *Let S_{NL} be a nonlinear subdivision scheme that reproduces constants and which can be written in the form (6), with S a linear subdivision scheme which is convergent. If F and δ satisfy the following two conditions,*

$$\exists M > 0 : \qquad ||F(d_1) - F(d_2)||_\infty \leq M||d_1 - d_2||_\infty, \tag{9}$$

$$\exists L > 0 \quad \exists T < 1 : \qquad ||\delta(S^L_{NL}(f) - S^L_{NL}(g))||_\infty \leq T||\delta(f-g)||_\infty, \tag{10}$$

$\forall f, g, d_1, d_2 \in l^\infty(\mathbb{Z})$, then the nonlinear subdivision scheme S_{NL} is stable.

Amat *et al.* use these results in [3] in order to obtain convergence and stability results for the $Power_p$ interpolatory subdivision schemes. In this paper, we use these results in order to study the convergence and stability properties of a new class of nonlinear interpolatory subdivision schemes based on a generalization of the $Power_p$ mean. We recall next certain basic facts about the $Power_p$ schemes.

3 $\boldsymbol{Power_p}$ Interpolatory Subdivision Schemes

The $Power_p$ schemes are introduced in [3,12], as a generalization of the *PPH* scheme in [4]. These schemes are defined as follows [3]

$$\begin{cases} (S_{Power_p} f)_{2n} = f_n, \\ (S_{Power_p} f)_{2n+1} = \frac{1}{2}(f_n + f_{n+1}) - \frac{1}{8} Power_p(\nabla^2 f_n, \nabla^2 f_{n+1}), \end{cases} \tag{11}$$

where the nonlinear mean

$$Power_p(x,y) = \frac{sign(x) + sign(y)}{2} \left| \frac{x+y}{2} \right| \left(1 - \left| \frac{x-y}{x+y} \right|^p \right), \tag{12}$$

is introduced in [18] in the context of nonlinear approximation techniques to the solution of hyperbolic conservation laws.

It is easy to see that $PPH(x,y) = Power_2(x,y)$. Moreover (see [18])

$$Power_1(x,y) = \frac{sign(x) + sign(y)}{2} \min\{|x|,|y|\} =: minmod(x,y).$$

4 $\boldsymbol{Weighted\text{-}Power_p}$ Interpolatory Subdivision Schemes

In this paper we propose to study the class of schemes that result from considering the following generalization of the $Power_p$ mean: For any $a > 0, b > 0$, we define the $Weighted\text{-}Power_p$ of two real numbers, x, y as follows,

$$WP_{p,a,b}(x,y) = \frac{sign(x) + sign(y)}{2} \left| \frac{ax+by}{a+b} \right| \left(1 - \frac{|x-y|^p}{(M + \frac{d}{c}m)(M + \frac{c}{d}m)^{p-1}} \right), \tag{13}$$

where $M = \max\{|x|,|y|\}$, $m = \min\{|x|,|y|\}$, $c = \max\{a,b\}$ and $d = \min\{a,b\}$.

It is easy to see that $WP_{p,\gamma,\gamma}(x,y) = Power_p(x,y)$, $\forall \gamma \in \mathbb{R}^+ \backslash \{0\}$. In addition,

$$WP_{p,a,b}(x,y) = WP_{p,\frac{a}{a+b},\frac{b}{a+b}}(x,y),$$

is also deduced directly from (13). Therefore we shall henceforth assume that the parameters a, b satisfy $0 < a, b < 1$ and $a + b = 1$.

The definition of the scheme $S_{WP_{p,a,b}}$ is, then, as follows

$$\begin{cases} (S_{WP_{p,a,b}} f)_{2n} = f_n, \\ (S_{WP_{p,a,b}} f)_{2n+1} = \frac{1}{2}(f_n + f_{n+1}) - \frac{1}{8} WP_{p,a,b}(\nabla^2 f_n, \nabla^2 f_{n+1}). \end{cases} \tag{14}$$

The properties of the $S_{WP_{p,a,b}}$ scheme are directly related to some properties of the $Weighted\text{-}Power_p$ mean, which are described next.

4.1 Properties of the *Weighted-Power$_p$* Mean

In what follows, and unless specifically stated, we shall assume that $a \geq 0, b \geq 0$, $a + b = 1$, $p \geq 1$. The following algebraic properties of the weighted average $WP_{p,a,b}$ are deduced directly from its definition (13).

Proposition 1. *Let $(x, y) \in \mathbb{R}^2$. The following properties hold*

$$\textbf{E1.} \qquad WP_{p,a,b}(x,x) = x. \tag{15}$$

$$\textbf{E2.} \qquad WP_{p,a,b}(x,y) = 0 \quad if \quad xy \leq 0. \tag{16}$$

$$\textbf{E3.} \qquad WP_{p,a,b}(x,y) = WP_{p,b,a}(y,x). \tag{17}$$

$$\textbf{E4.} \qquad WP_{p,a,b}(-x,-y) = -WP_{p,a,b}(x,y). \tag{18}$$

The following proposition generalizes a key result in [3]. It is a consequence of an additive property, similar to the one obtained for the *Power$_p$* mean.

Proposition 2. *Let $\alpha = c/d = \max\{a,b\}/\min\{a,b\}$. Then for each $p \geq 1$, the* Weighted-Power$_p$ *mean satisfies the following relation*

$$WP_{p,a,b}(x,y) = \frac{sign(x) + sign(y)}{2} \frac{ax + by}{cM + dm} m \left[1 + \alpha \sum_{k=1}^{p-1} \frac{|x-y|^k}{(M + \alpha m)^k}\right]. \tag{19}$$

Proof. Let us asume that $x, y > 0$, since all other cases are either trivial or can be reduced to this one using (16) and (18). The result can be deduced from exploting the relation between the *Weighted-Power$_p$* means corresponding to two consecutive indices, say p and $p + 1$. Notice that

$$\begin{aligned} WP_{p+1,a,b}(x,y) &= (ax+by)\left(1 - \frac{|x-y|^{p+1}}{(M + \frac{d}{c}m)(M + \frac{c}{d}m)^p}\right) \\ &= (ax+by)\left(1 - \frac{|x-y|^p}{(M + \frac{1}{\alpha}m)(M+\alpha m)^{p-1}} \frac{|x-y|}{(M+\alpha m)}\right). \end{aligned}$$

Since

$$\begin{matrix} M = \frac{1}{2}(|x+y| + |x-y|) & \multirow{2}{*}{} & |x+y| = M + m \\ m = \frac{1}{2}(|x+y| - |x-y|) & i.e. & |x-y| = M - m, \end{matrix} \tag{20}$$

we have that

$$\frac{|x-y|}{M+\alpha m} = \frac{M-m}{M+\alpha m} = 1 - (1+\alpha)\frac{m}{M+\alpha m}, \tag{21}$$

so that we can write

$$\begin{aligned} WP_{p+1,a,b}(x,y) &= WP_{p,a,b}(x,y) + (1+\alpha)m \frac{ax+by}{(M + \frac{1}{\alpha}m)} \frac{|x-y|^p}{(M+\alpha m)^p} \\ &= WP_{p,a,b}(x,y) + \alpha m \frac{ax+by}{(cM+dm)} \frac{|x-y|^p}{(M+\alpha m)^p}, \end{aligned}$$

where we have used that

$$c(\alpha+1) = c\frac{c+d}{d} = \frac{c}{d} = \alpha.$$

On the other hand, it is easy to see that

$$\frac{|x-y|}{M+\frac{m}{\alpha}} = \frac{M-m}{M+\frac{m}{\alpha}} = 1-\left(1+\frac{1}{\alpha}\right)\frac{m}{M+\frac{m}{\alpha}} = 1-\frac{m}{c(M+\frac{m}{\alpha})}, \tag{22}$$

from which we readily deduce

$$WP_{1,a,b}(x,y) = (ax+by)\left(1-\frac{|x-y|}{M+\frac{1}{\alpha}m}\right) = \frac{ax+by}{cM+dm}m.$$

This allows us to prove (19). □

The next result states certain estimates for the *Weighted- Power*$_p$ mean which generalize those in [3] for the *Power*$_p$ mean.

Proposition 3. *The mean* $\mathrm{WP}_{p,a,b}$ *with* $p \geq 1$ *satisfies:*

C1 $|WP_{p,a,b}(x,y)| \leq \max\{|x|,|y|\}, \quad \forall (x,y) \in \mathbb{R}^2.$ (23)

C2 $|WP_{p,a,b}(x,y)| \leq 2\max\{|ax|,|by|\}, \quad \forall (x,y) \in \mathbb{R}^2.$ (24)

C3 $|WP_{p,a,b}(x,y)| \leq p\alpha\min\{|x|,|y|\}, \quad \forall (x,y) \in \mathbb{R}^2.$ (25)

C4 $\forall x,y>0, \quad \forall q>p, WP_{p,a,b}(x,y) \leq WP_{q,a,b}(x,y),$ *and*

$$\frac{1}{\alpha}\min\{x,y\} \leq \frac{ax+by}{cM+dm}\min\{x,y\} \leq WP_{p,a,b}(x,y) \leq ax+by. \tag{26}$$

where $\alpha = c/d = \max\{a,b\}/\min\{a,b\}$.

Proof. As in the case of the previous proposition, it is enough to consider the case $x,y>0$ in order to prove **C1**, **C2** and **C3**.

C1: From (21) we readily deduce that

$$\frac{|x-y|}{M+\alpha m} = 1-\frac{m}{dM+cm},$$

and since

$$0 < \frac{m}{dM+cm} = \frac{(d+c)m}{dM+cm} < 1,$$

we readily get that

$$0 < \frac{|x-y|}{M+\alpha m} < 1.$$

Similarly, using (22) we get

$$0 < \frac{|x-y|}{M+\frac{m}{\alpha}} < 1,$$

hence

$$0 < 1 - \frac{|x-y|^p}{(M+\frac{1}{\alpha}m)(M+\alpha m)^{p-1}} < 1.$$

Thus,

$$\begin{aligned}|WP_{p,a,b}(x,y)| &\leq |(ax+by)| \left| \left(1 - \frac{|x-y|^p}{(M+\frac{1}{\alpha}m)(M+\alpha m)^{p-1}} \right) \right| \leq |ax+by| \\ &\leq \max\{|x|,|y|\},\end{aligned}$$

since $a+b=1$.

C2: Proceeding as before, but using $|ax+by| \leq |ax|+|by|$ in the last step of the estimations above, we get

$$|WP_{p,a,b}(x,y)| \leq |ax+by| \leq 2\max\{|ax|,|by|\}.$$

C3: We use equation (19). We have seen that $|x-y|/(M+\alpha m) \leq 1$. In addition, it is easy to deduce that $(ax+by)/(cM+dm) \leq 1$. Hence

$$1+\alpha\sum_{k=1}^{p-1}\frac{|x-y|^k}{(M+\alpha m)^k} \leq (1+\alpha(p-1)) \leq p\alpha,$$

since $1-\alpha < 0$. The desired estimate follows easily.

C4: This follows in a straightforward manner from relation (19).

In the following section we will see that these properties lead to a proof of the convergence of the $S_{WP_{p,a,b}}$ schemes, based on Theorem 1.

4.2 Convergence Analysis

Notice that the scheme $S_{WP_{p,a,b}}$ in (14) can be written as a nonlinear perturbation of the $S_{1,1}$ scheme, which is C^{1-} convergent. The perturbation is given by the operator $F(d)$ so that $F(d)_{2n+1} = -\frac{1}{8}WP_{p,a,b}(d_n,d_{n+1})$, and the linear operator δ is $(\delta f)_n = \nabla^2 f_n = f_{n+1} - 2f_n + f_{n-1}$. The properties of the *Weighted-Power$_p$* mean shown in the previous section will allow us to generalize the convergence results in [3,12] for the *Power$_p$* schemes.

Theorem 3. *For all $p \geq 1$, the nonlinear subdivision scheme $S_{WP_{p,a,b}}$ is uniformly convergent, and for any initial sequence the limit function belongs to C^{1-}.*

Proof. The necessary estimates on the norm of the nonlinear operator F are derived from the estimates for the *Weighted-Power$_p$* mean. Using (23), we get for $d \in l^\infty(\mathbb{Z})$

$$|(F(d))_{2n+1}| \leq \frac{1}{8}\max\{|d_n|,|d_{n+1}|\}, \quad \forall n \quad \rightarrow \quad ||F(d)||_\infty \leq \frac{1}{8}||d||_\infty. \tag{27}$$

In order to check (8), we consider separately the case of odd and even indices.

$$\begin{aligned}(\nabla^2(S_{WP_{p,a,b}}f))_{2n+1} &= f_n - 2(S_{WP_{p,a,b}}f)_{2n+1} + f_{n+1}\\ &= \frac{1}{4}WP_{p,a,b}(\nabla^2 f_n, \nabla^2 f_{n+1}).\end{aligned}$$

Hence, from (23), we get

$$|(\nabla^2(S_{WP_{p,a,b}}f))_{2n+1}| \leq \frac{1}{4}||\nabla^2 f||_\infty. \tag{28}$$

For even components we have instead

$$\begin{aligned}(\nabla^2(S_{WP_{p,a,b}}f))_{2n} &= (S_{WP_{p,a,b}}f)_{2n+1} - 2f_n + (S_{WP_{p,a,b}}f)_{2n-1}\\ &= \frac{\nabla^2 f_n}{2} - \frac{1}{8}(WP_{p,a,b}(\nabla^2 f_n, \nabla^2 f_{n+1}) + WP_{p,a,b}(\nabla^2 f_{n-1}, \nabla^2 f_n)).\end{aligned}$$

From these relations and (23) we easily get that

$$|(\nabla^2(S_{WP_{p,a,b}}f))_{2n}| \leq \frac{3}{4}||\nabla^2 f||_\infty,$$

hence $||\nabla^2 S_{WP_{p,a,b}}f||_\infty \leq \frac{3}{4}||\nabla^2 f||_\infty$ and, from Theorem 1, we have $S_{WP_{p,a,b}}f$ is $C^{\beta-}$ convergent with $\beta = -log_2(3/4)$. However, the bound above can be set to $1/2$ by using the same trick as in [6]. Define the function

$$Z_{p,a,b}(x,y,z) = \frac{x}{2} - \frac{1}{8}(WP_{p,a,b}(x,y) + WP_{p,a,b}(z,x)).$$

Then, following the same steps as in [12]-Theorem II.7, we have $|Z_{p,a,b}(x,y,z)| \leq \frac{1}{2}\max\{|x|,|y|,|z|\}$. Hence,

$$|(\nabla^2(S_{WP_{p,a,b}}f))_{2n}| \leq \frac{1}{2}||\nabla^2 f||_\infty, \tag{29}$$

and

$$||\nabla^2(S_{WP_{p,a,b}}f)||_\infty \leq \frac{1}{2}||\nabla^2 f||_\infty \quad \forall p.$$

This estimate, allows us to conclude that the $WP_{p,a,b}$ schemes are, in fact, at least C^{1-} convergent.

4.3 Stability Analysis

As in the case of the *Power*$_p$ schemes, stability follows from certain estimates on the difference of two weighted harmonic means. We have only been able to obtain the necessary estimates for $p = 2$. We notice that for $p = 2$, the $WP_{p,a,b}$ mean takes the following simpler form.

Proposition 4

$$WP_{2,a,b}(x,y) = \frac{sign(x) + sign(y)}{2}\frac{xy}{a|y| + b|x|}. \tag{30}$$

Proof. We first notice that

$$\left(M+\frac{d}{c}m\right)\left(M+\frac{c}{d}m\right)=M^2+m^2+\left(\frac{d}{c}+\frac{c}{d}\right)Mm=(M-m)^2+\frac{Mm}{cd}$$
$$=(x-y)^2+\frac{|x||y|}{ab},$$

since $Mm=|x||y|$, $c+d=a+b=1$ and $cd=ab$.

Let us assume that $x>0, y>0$ since all other cases are either trivial or can be reduced to this one, by using the properties of the $WP_{p,a,b}$ mean. Then

$$1-\frac{(x-y)^2}{(M+\frac{d}{c}m)(M+\frac{c}{d}m)}=1-\frac{(x-y)^2}{(x-y)^2+\frac{xy}{ab}}=\frac{\frac{xy}{ab}}{(x-y)^2+\frac{xy}{ab}}.$$

But we can also write

$$(x-y)^2+\frac{xy}{ab}=x^2+y^2+\left(\frac{a}{b}+\frac{b}{a}\right)xy=\frac{1}{ab}(ax+by)(ay+bx).$$

Hence, substituting into the definition of $WP_{2,a,b}$ we have

$$WP_{2,a,b}(x,y)=(ax+by)\left(1-\frac{(x-y)^2}{(M+\frac{d}{c}m)(M+\frac{c}{d}m)}\right)=\frac{xy}{bx+ay}.\quad\square$$

These relations become useful in proving the following proposition.

Proposition 5. $\forall(x_1,y_1),(x_2,y_2)\in\mathbb{R}^2$

$$|WP_{2,a,b}(x_1,y_1)-WP_{2,a,b}(x_2,y_2)|\le 2\alpha\max\{|x_1-x_2|,|y_1-y_2|\}. \tag{31}$$

Proof. We consider the following cases:

1. The cases $x_1y_1\le 0$ and $x_2y_2\le 0$ are trivial since the left hand side (l.h.s) of (31) is zero.
2. If $x_1y_1>0$ and $x_2y_2\le 0$, we get

$$|WP_{p,a,b}(x_1,y_1)-WP_{p,a,b}(x_2,y_2)|=|WP_{p,a,b}(x_1,y_1)|\le p\alpha\min\{|x_1|,|y_1|\}.$$

   Then, given the sign combination between the points, we can have only two situations: Either $x_1x_2<0$, in which case

$$\begin{aligned}|WP_{p,a,b}(x_1,y_1)-WP_{p,a,b}(x_2,y_2)|&\le p\alpha|x_1|\le p\alpha|x_1-x_2|\\&\le p\alpha\max\{|x_1-x_2|,|y_1-y_2|\},\end{aligned}$$

   or $y_1y_2<0$, which is absolutely similar.
3. The case $x_1y_1\le 0$ and $x_2y_2>0$ is equivalent to the previous one.
4. Let us assume now that $x_1y_1>0$ and $y_2x_2>0$. Then either (x_1,y_1) and (x_2,y_2) both belong to the first or the third quadrant, i.e. $x_1x_2>0$ and $y_1y_2>0$, or to opposite quadrants, in which case $x_1x_2<0$ and $y_1y_2<0$.

(a) If $x_1x_2 < 0$ and $y_1y_2 < 0$ the two points belong to opposite quadrants of the plane. Then, we have from equation (25):

$$\begin{aligned}|WP_{p,a,b}(x_1,y_1) - WP_{p,a,b}(x_2,y_2)| &\le p\alpha(\min\{|x_1|,|y_1|\} + \min\{|x_2|,|y_2|\}) \\ &\le p\alpha \max\{|x_1-x_2|,|y_1-y_2|\}.\end{aligned}$$

(b) If $x_1x_2 > 0$ and $y_1y_2 > 0$ both points belong to the same quadrant. Without loss of generality, we can assume $x_{1,2} > 0, y_{1,2} > 0$.
Using (30) we have

$$WP_{2,a,b}(x,y) = \frac{xy}{bx+ay}.$$

Applying the mean value theorem, we have

$$\begin{aligned}&|WP_{2,a,b}(x_1,y_1) - WP_{2,a,b}(x_2,y_2)| \\ &\qquad \le ||\nabla WP_{2,a,b}(\xi_x,\xi_y)||_\infty ||(x_1,y_1)-(x_2,y_2)||_\infty,\end{aligned}$$

so that we only need to bound

$$\partial_x WP_{2,a,b}(x,y) = \frac{ay^2}{(ay+bx)^2}, \qquad \partial_y WP_{2,a,b}(x,y) = \frac{bx^2}{(ay+bx)^2}.$$

We can easily prove that

$$0 \le \partial_x WP_{2,a,b}(x,y) \le \frac{1}{a}, \qquad 0 \le \partial_y WP_{2,a,b}(x,y) \le \frac{1}{b},$$

hence

$$||\nabla WP_{2,a,b}(\xi_x,\xi_y)||_\infty \le \frac{1}{d},$$

and

$$|WP_{2,a,b}(x_1,y_1) - WP_{2,a,b}(x_2,y_2)| \le \frac{1}{d}\max\{|x_1-x_2|,|y_1-y_2|\}.$$

Since $1/d < 2\alpha$, the bound (31) holds in all cases. □

As in [6], the stability of the *Weighted-Power*$_p$ schemes for $p = 2$ follows from Theorem 2, thanks to a two step contraction property, analogous to the one shown in [6] for the *PPH* scheme.

Proposition 6. *If* $ab \ge \frac{1}{8}$, $\hat f = S_{WP_{2,a,b}}(f)$, $\hat g = S_{WP_{2,a,b}}(g)$, $\bar f = S_{WP_{2,a,b}}(\hat f)$, $\bar g = S_{WP_{2,a,b}}(\hat g)$, *then*

1 $||\nabla^2 \hat f||_\infty \le \frac{1}{2}||\nabla^2 f||_\infty$.

2. $|\nabla^2(\hat f_{2n+1} - \hat g_{2n+1})| \le \frac{\alpha}{2}||\nabla^2(f-g)||_\infty$.

$$|\nabla^2(\hat f_{2n} - \hat g_{2n})| \le \left(\frac{1}{2} + \frac{\alpha}{2}\right)||\nabla^2(f-g)||_\infty.$$

3. $||\nabla^2(\bar f - \bar g)||_\infty \le \tau ||\nabla^2(f-g)||_\infty, \quad \tau = \max\left\{\frac{1+\alpha+\alpha^2}{4}, \frac{2\alpha+\alpha^2}{4}\right\},$

where $\alpha = c/d = \max\{a,b\}/\min\{a,b\}$.

The proof follows the same steps in [6] and is omitted. As in [6], it requires certain bounds on an auxiliary function $Z_{2,a,b}(x,y,z)$, which are provided by the following lemma. We include the proof of the lemma, because it serves to show that the lack of symmetry of the $WP_{2,a,b}$-mean leads to certain constraints in the coefficients, in order to ensure analogous stability bounds.

Lemma 1. *Let $Z_{2,a,b}:\mathbb{R}^3\to\mathbb{R}$ be defined as*

$$Z_{2,a,b}(x,y,z)=\frac{x}{2}-\frac{1}{8}(WP_{2,a,b}(x,y)+WP_{2,a,b}(z,x)), \tag{32}$$

then, provided $ab\geq 1/8$, the following two inequalities hold:

$$\textbf{1}\quad |Z_{2,a,b}(x,y,z)|\leq\frac{|x|}{2}, \tag{33}$$

$$\begin{aligned}\textbf{2}\quad &|Z_{2,a,b}(x_1,y_1,z_1)-Z_{2,a,b}(x_2,y_2,z_2)|\\ &\qquad\leq\frac{1}{2}|x_1-x_2|+\frac{\alpha}{2}\max\{|y_1-y_2|,|z_1-z_2|\}.\end{aligned} \tag{34}$$

Proof. To prove (33), we use (30) and

$$0\leq\frac{sign(x)+sign(y)}{2}\frac{y}{a|y|+b|x|}\leq\frac{1}{a},\ 0\leq\frac{sign(z)+sign(x)}{2}\frac{z}{a|x|+b|z|}\leq\frac{1}{b}.$$

Then,

$$\left|1-\frac{1}{4}\frac{sign(x)+sign(y)}{2}\frac{y}{a|y|+b|x|}-\frac{1}{4}\frac{sign(z)+sign(x)}{2}\frac{z}{a|x|+b|z|}\right|\leq 1,$$

provided $ab\geq 1/8$, which proves (1).

Let us prove (34). We denote $Z_i:=Z_{2,a,b}(x_i,y_i,z_i)$, $i=1,2$. Suppose first that $x_1\geq 0$ and $x_2\leq 0$. Then

$$|Z_1-Z_2|\leq|Z_1|+|Z_2|\leq\frac{|x_1|}{2}+\frac{|x_2|}{2}=\frac{x_1-x_2}{2}=\frac{|x_1-x_2|}{2}.$$

Let us assume now that $x_1>0$, $x_2>0$. We can write

$$Z_1-Z_2=\frac{x_1-x_2}{2}-\frac{1}{8}B-\frac{1}{8}C,$$

where

$$B=WP_{2,a,b}(x_1,y_1)-WP_{2,a,b}(x_2,y_2),\quad C=WP_{2,a,b}(z_1,x_1)-WP_{2,a,b}(z_2,x_2).$$

Then, if $y_1>0$, $y_2>0$ we can write

$$B=\frac{x_1y_1}{(bx_1+ay_1)}-\frac{x_2y_2}{(bx_2+ay_2)}=B'+B'',$$

where

$$B' = \frac{bx_1x_2}{(bx_1+ay_1)(bx_2+ay_2)}(y_1-y_2), \quad B'' = \frac{ay_1y_2}{(bx_1+ay_1)(bx_2+ay_2)}(x_1-x_2).$$

Likewise, if $z_1 > 0$, $z_2 > 0$, we can write

$$C = \frac{x_1z_1}{(bz_1+ax_1)} - \frac{x_2z_2}{(bz_2+ax_2)} = C' + C'',$$

where

$$C' = \frac{ax_1x_2}{(ax_1+bz_1)(ax_2+bz_2)}(z_1-z_2), \quad C'' = \frac{bz_1z_2}{(ax_1+bz_1)(ax_2+bz_2)}(x_1-x_2).$$

It can readily be observed that, since all variables involved have the same sign,

$$|B'| \le \frac{1}{b}|y_1-y_2|, \qquad |C'| \le \frac{1}{a}|z_1-z_2|.$$

For the terms B'' and C'', we notice that

$$\begin{aligned}&\frac{x_1-x_2}{2} - \frac{1}{8}B'' - \frac{1}{8}C'' \\ &\quad = \left(\frac{x_1-x_2}{2}\right)\left(1 - \frac{ay_1y_2}{4(bx_1+ay_1)(bx_2+ay_2)} - \frac{bz_1z_2}{4(ax_1+bz_1)(ax_2+bz_2)}\right).\end{aligned}$$

Since all variables involved have the same sign, we have that

$$0 \le \frac{ay_1y_2}{(bx_1+ay_1)(bx_2+ay_2)} \le \frac{1}{a}, \quad 0 \le \frac{bz_1z_2}{(ax_1+bz_1)(ax_2+bz_2)} \le \frac{1}{b},$$

from which we can easily deduce that

$$\left|1 - \frac{ay_1y_2}{4(bx_1+ay_1)(bx_2+ay_2)} - \frac{bz_1z_2}{4(ax_1+bz_1)(ax_2+bz_2)}\right| \le 1,$$

provided $ab \ge 1/8$.

Summarizing, if all variables involved are positive,

$$\begin{aligned}|Z_1 - Z_2| &\le \left|\frac{x_1-x_2}{2} - \frac{1}{8}B'' - \frac{1}{8}C''\right| + \frac{|y_1-y_2|}{8b} + \frac{|z_1-z_2|}{8a} \\ &\le \frac{1}{2}|x_1-x_2| + \frac{1}{8ab}\max\{|y_1-y_2|, |z_1-z_2|\},\end{aligned}$$

provided that $ab \ge 1/8$.

Let us assume now that $y_1y_2 < 0$, and, without loss of generality, assume $y_1 > 0$, $y_2 < 0$. Then, $B = WP_{2,a,b}(x_1, y_1)$, so that **C3** gives $|B| \le 2\alpha \min\{x_1, y_1\} \le 2\alpha|y_1 - y_2|$. Analogously, if $z_1z_2 < 0$, we deduce that $|C| \le 2\alpha|z_1 - z_2|$.

Then, if $y_1 \cdot y_2 < 0$ and $z_1 > 0, z_2 > 0$ we can write

$$\begin{aligned}|Z_1 - Z_2| &\le \left|\frac{x_1 - x_2}{2} - \frac{1}{8}C''\right| + \frac{1}{8}|B| + \frac{1}{8}|C'| \\ &\le \left|\frac{x_1 - x_2}{2} - \frac{1}{8}C''\right| + \frac{1}{8}\left(2\alpha|y_1 - y_2| + \frac{1}{a}|z_1 - z_2|\right).\end{aligned}$$

To bound the first term, we proceed as before. In this case, we can easily prove that

$$\left|1 - \frac{bz_1z_2}{4(ax_1 + bz_1)(ax_2 + bz_2)}\right| \le 1,$$

provided $b \ge 1/8$. Hence

$$|Z_1 - Z_2| \le \left|\frac{x_1 - x_2}{2}\right| + \frac{1}{8}\left(2\alpha + \frac{1}{a}\right)\max\{|y_1 - y_2|, |z_1 - z_2|\},$$

provided $b \ge 1/8$.

A similar argument allows us to prove that if $y_1 > 0, y_2 > 0$ and $z_1 \cdot z_2 < 0$ we get

$$|Z_1 - Z_2| \le \left|\frac{x_1 - x_2}{2}\right| + \frac{1}{8}\left(\frac{1}{a} + 2\alpha\right)\max\{|y_1 - y_2|, |z_1 - z_2|\},$$

provided $a \ge 1/8$.

Lastly, if $y_1 \cdot y_2 < 0$ and $z_1 \cdot z_2 < 0$, we have

$$\begin{aligned}|Z_1 - Z_2| &\le \left|\frac{x_1 - x_2}{2}\right| + \frac{1}{8}|B| + \frac{1}{8}|C| \\ &\le \left|\frac{x_1 - x_2}{2}\right| + \frac{2\alpha}{8}\left(|y_1 - y_2| + |z_1 - z_2|\right) \\ &\le \left|\frac{x_1 - x_2}{2}\right| + \frac{\alpha}{2}\max\{|y_1 - y_2|, |z_1 - z_2|\}.\end{aligned}$$

Since $ab \ge 1/8$ implies $a \ge 1/8$ and $b \ge 1/8$, we conclude that

$$|Z_1 - Z_2| \le \frac{1}{2}|x_1 - x_2| + \rho\max\{|y_1 - y_2|, |z_1 - z_2|\},$$

with $\rho = \frac{1}{8}\max\{\frac{1}{ab}, 2\alpha + \frac{1}{d}, 4\alpha\} = \alpha/2$, which concludes the proof. □

It is worth to remark that $WP_{2,.5,.5}(x, y) = PPH(x, y)$, and that in this case $\alpha = 1$ and we recover *exactly* the same bounds found in [6] for the *PPH* scheme.

The previous result leads to the following

Theorem 4. *If $ab \ge \frac{1}{8}$ and $\alpha = \frac{\max\{a,b\}}{\min\{a,b\}} \le -1 + \sqrt{5}$ then the scheme $S_{WP_{2,a,b}}$ is stable.*

Proof. Since $\alpha > 1$, $\tau = \max\left\{\frac{1+\alpha+\alpha^2}{4}, \frac{2\alpha+\alpha^2}{4}\right\} = \frac{\alpha}{2} + (\frac{\alpha}{2})^2$. Then $0 < \tau < 1$ if and only if $\alpha < -1 + \sqrt{5}$.

4.4 Order of Approximation

In this section we study the order of approximation of the $WP_{p,a,b}$ schemes. For this task we rely on the following definitions and results, which can be found in [12].

Definition 4. Polynomial Reproduction. *A subdivision scheme S is said to reproduce* ***exactly*** *polynomials of degree k if for all polynomial, P, of degree at most k the following condition holds true: if $f_n = P(n)$, $n \in \mathbb{Z}$, then $(Sf)_n = P(2^{-1}n)$, $n \in \mathbb{Z}$.*

Definition 5. Order of Approximation. *A subdivision scheme S is said to have an order k of approximation if for any function $g \in C^k$ the following condition holds true:*

$$\textit{if} \quad f = g(h\,\cdot), \quad \textit{then} \quad |Sf - g(2^{-1}h\,\cdot)| \le Ch^k, \quad \forall h > 0.$$

Definition 6. *Let S be a convergent subdivision scheme. We say that S^∞ has order of approximation equal to k if for any function $g \in C^k$ and any $h > 0$ we have*

$$||S^\infty f - g||_\infty \le Ch^k, \qquad f = \{g(nh)\}_{n\in\mathbb{Z}}, \tag{35}$$

where C is a constant which is independent of h but may depend on g.

Proposition 7. *If a scheme S is stable and reproduces exactly polynomials of degree k then S has order of approximation equal to $k+1$.*

The reader is referred to [11] for a recent contribution to the issue of the more general concept of polynomial generation versus exact polynomial reproduction, which are equivalent for interpolatory schemes.

Theorem 5. *Let S_{NL} be a convergent scheme and S a linear convergent scheme which has order of approximation equal to k, and such that $\forall f_0 = g(nh)_n$ $g \in C^\infty([0,1])$*

$$||S_{NL}f_0 - Sf_0|| = O(h^r). \tag{36}$$

Then S_{NL} has order of approximation equal to $\min\{r, k\}$.

The following lemma provides straightforward generalizations of analogous results for the $Power_p$ schemes.

Lemma 2. *For all p,a,b, the scheme $S_{WP_{p,a,b}}$ reproduces exactly polynomials of degree 2. In addition,*

$$||S_{WP_{p,a,b}}f - S_{1,1}f||_\infty = O(h^2). \tag{37}$$

Proof. To prove the polynomial reproduction property, we take $P(x) = Ax^2 + Bx + C$. Then $(\nabla^2 P)_n = 2A$ $\forall n \in \mathbb{Z}$, and $WP_{p,a,b}((\nabla^2 P)_n, (\nabla^2 P)_{n+1}) = 2A = \text{mean}((\nabla^2 P)_n, (\nabla^2 P)_{n+1})$ $\forall n \in \mathbb{Z}$, from (13). Hence

$$(S_{WP_{p,a,b}}f)_{2n+1} = (S_{2,2}P)_{2n+1},$$

and we obtain the result from the polynomial reproduction property of the $S_{2,2}$ scheme.

To prove (37), we consider $g \in C^\infty([0,1])$ and $f_n = g(nh)\;\; n \in \mathbb{Z}$ and $h > 0$. By Taylor's expansion, we get that $\nabla^2 f_n = O(h^2)$, $\forall n$. Using (23) we get

$$(S_{WP_{p,a,b}} f)_{2n+1} - (S_{1,1} f)_{2n+1} = -\frac{1}{8} WP_{p,a,b}(\nabla^2 f_n, \nabla^2 f_{n+1}) = O(h^2).$$

Lemma 2 and Theorem 5 allow us to state the following general result.

Proposition 8. *The* $\mathrm{WP}_{p,a,b}$ *schemes have order of approximation equal to 2.*

On the other hand, in the previous section we have proven the stability of the $WP_{2,a,b}$ schemes, under certain conditions on the coefficients a, b. Then, Lemma 2 and Proposition 7 lead to the following result

Proposition 9. *If the* $\mathrm{WP}_{2,a,b}$ *scheme is stable, then it has order of approximation equal to 3.*

Remark: We notice that the *Power*$_2$ schemes have order of approximation equal to 4 (see [12]). The results in this section can be interpreted as follows: if a and b are not $1/2$, the $WP_{2,a,b}$ scheme defined in (14) only represents a weighted mean of schemes that are third order accurate, which is in turn third order accurate. Fourth order accuracy can only be obtained for the choice $a = 1/2$, $b = 1/2$. Similarly, for the $S_{3,2}$ scheme described in Section 2, it will be necessary to consider the coeficients $a = 3/8$, $b = 5/8$ in order to attain maximum accuracy.

5 Numerical Examples

In this section we shall present several numerical experiments in order to illustrate the non-oscillatory properties of the $WP_{p,a,b}$ mean and the influence of the parameters in the proposed nonlinear schemes.

The initial data are obtained from the function $f(x)$

$$f(x) = \begin{cases} \sin(\pi x), & for \quad x \in [0, 0.5] \\ -\sin(\pi x), & for \quad x \in]0.5, 1], \end{cases}$$

over the mesh $x = 0 : 0.1 : 1$.

In Figure 1 we show the result of three applications of the $S_{2,1}$, $S_{2,2}$ and $WP_{p,1/2,1/2}$ for $p = 1, 2$, i.e. the *Power*$_p$ schemes with $p = 1, 2$. It can be clearly appreciated in these figures that the *Power*$_p$ mean forces the scheme to behave as the $S_{2,1}$ scheme to the left of the discontinuity, and as the symmetric $S_{1,2}$ scheme to the right of the discontinuity. At the discontinuity, it behaves as the $S_{1,1}$ scheme. The behavior of the *Power*$_p$ schemes for $p = 1$ and $p = 2$ is quite similar for the discrete data considered.

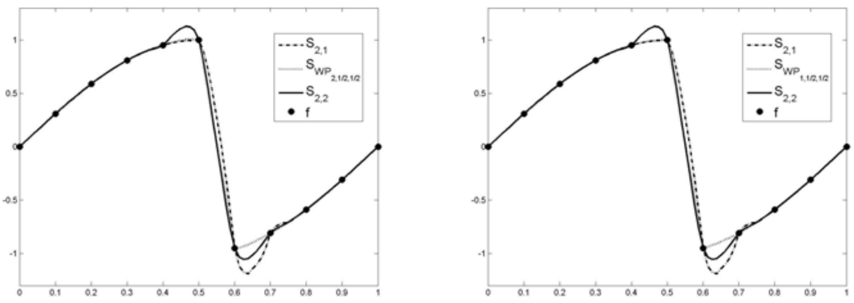


Fig. 1. (●) Initial data, results after 3 applications of $S_{2,2}$ (solid line), $S_{2,1}$ (dashdot line) and $S_{WP_{2,0.5,0.5}}$ (dotted line) on the left plot and $S_{WP_{1,0.5,0.5}}$ (dotted line) on the right plot

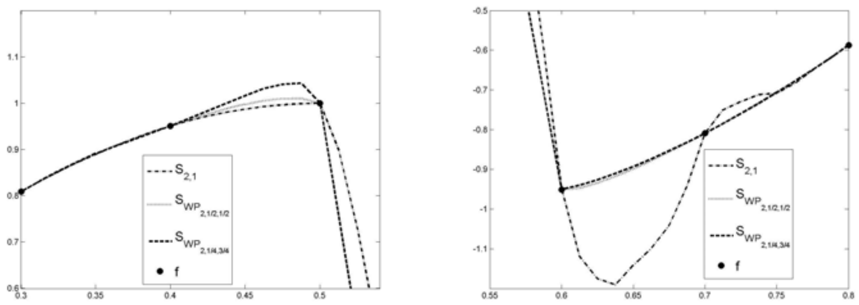


Fig. 2. (●) Initial data. Zoom of results after 3 applications of $S_{2,1}$ (dashdot line), $S_{WP_{2,0.5,0.5}}$ (dotted line) and $S_{WP_{2,0.25,0.75}}$ (dashed line)

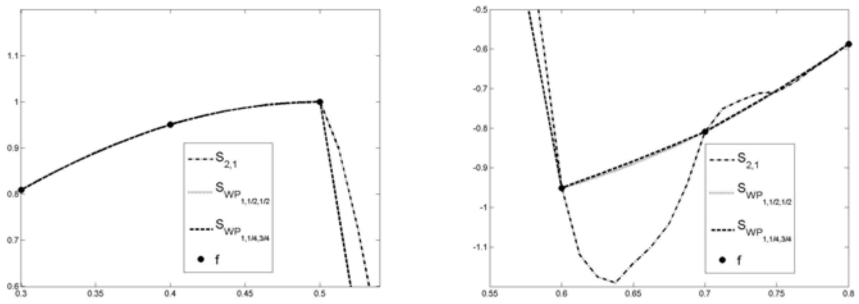


Fig. 3. (●) Initial data. Zoom of results after 3 applications of $S_{2,1}$ (dashdot line), $S_{WP_{1,0.5,0.5}}$ (dotted line) and $S_{WP_{1,0.25,0.75}}$ (dashed line)

In Figures 2 ($p = 2$) and 3 ($p = 1$) we show a close-up view of the region around the discontinuity after 3 applications of the $WP_{p,a,b}$ schemes with $a = b = 1/2$ and $a = 1/4$, $b = 3/4$, together with the $S_{2,1}$ scheme. We have shown in the previous sections that we can only ensure third order accuracy for the $WP_{2,.25,.75}$, provided the scheme is stable. We see that $ab = 3/16 > 1/8$, but $\max(a,b)/\min(a,b) = 3 > \sqrt{5} - 1$, so that the stability of this scheme cannot be ensured. The fact that $a < b$ seems to lead to a larger influence of the $S_{1,2}$ scheme, while still producing essentially non-oscillatory results.

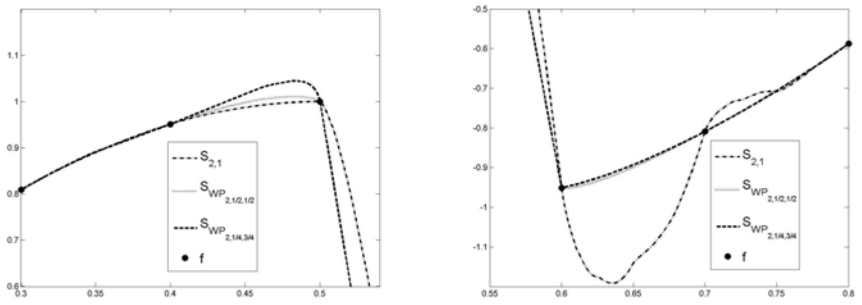


Fig. 4. (●) Initial data. Zoom of results after 7 applications of $S_{2,1}$ (dashdot line), $S_{WP_{2,0.5,0.5}}$ (dotted line) and $S_{WP_{2,0.25,0.75}}$ (dashed line)

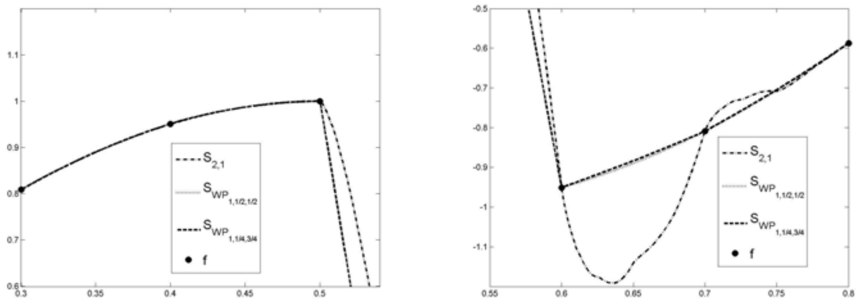


Fig. 5. (●) Initial data. Zoom of results after 7 applications of $S_{2,1}$ (dashdot line), $S_{WP_{1,0.5,0.5}}$ (dotted line) and $S_{WP_{1,0.25,0.75}}$ (dashed line)

In Figures 5 and 4 we show the outcome of the $WP_{p,a,b}$ for $p = 2$ (left) and $p = 1$ (right), for $a = 1/4$, $b = 3/4$, and $a = b = 1/2$, after 7 applications of the scheme. We can see that there is no noticeable difference between the results after 3 or 7 applications, hence the data shown can be considered as the limit function.

6 Conclusion

We have introduced a weighted version of the *Power*$_p$ mean, called *Weighted-Power*$_p$ mean and have used it to define a class of nonlinear interpolatory subdivision schemes that generalize the *Power*$_p$ schemes considered in [12]. Our analysis shows that the $WP_{p,a,b}$ mean shares many desirable properties with the *Power*$_p$ mean, and that the new schemes are also non-oscillatory around discrete data with large gradientes.

Clearly, the new schemes represent an improvement over the linear, 4-point, $S_{2,2}$ scheme. However, both the theoretical results and the numerical experiments reveal that there is no improvement with respect to the symmetric case, $a = b = 1/2$, i.e. the *Power*$_p$ schemes. The choice of parameters $a = b = 1/2$ is optimal from the point of view of the accuracy of the scheme, and the stability of the resulting schemes can only be ensured if the parameters are very close to that value.

On the other hand, we expect that the ability to consider weighted harmonic means can be of use in constructing nonlinear versions of other linear schemes, of higher order of approximation. In particular, the nonlinear scheme given as

$$\begin{cases} (\hat{S}_{3,2}f)_{2n} = f_n, \\ (\hat{S}_{3,2}f)_{2n+1} = \frac{1}{2}(f_n + f_{n+1}) - \frac{1}{16}WP_{p,\frac{3}{8},\frac{5}{8}}\left(3\nabla^2 f_n - \nabla^2 f_{n-1}, \nabla^2 f_n - \nabla^2 f_{n+1}\right) \end{cases} \tag{38}$$

will be fifth order accurate. We also expect that the tools developed in this paper can serve to analyze the convergence and stability of this scheme. Ultimately, we want to develop a nonlinear version of the 6-point DD interpolatory scheme with high order accuracy and a non-oscillatory behavior around discontinuities. This is currently work in progress.

Acknowledgments. Maria Santágueda enjoys a scholarship granted for the Ministerio de Educación y Ciencia with reference $AP2006-04068$. The authors are grateful for the funding of the project $MTM2008-00974$.

References

1. Amat, S., Dadourian, K., Donat, R., Liandrat, J., Trillo, J.C.: Error bounds for a convexity-preserving interpolation and its limit function. J. Comput. Appl. Math. 211(1), 36–44 (2008)
2. Amat, S., Dadourian, K., Liandrat, J.: On a nonlinear subdivision scheme avoiding Gibbs oscillations and converging towards C^s- functions with $s > 1$. Math. Comp. 80, 959–971 (2011)
3. Amat, S., Dadourian, K., Liandrat, J.: Analysis of a class of nonlinear subdivision schemes and associated multiresolution transforms. Adv. Comput. Math. 80, 253–277 (2011)
4. Amat, S., Donat, R., Liandrat, J., Trillo, J.C.: Analysis of a class of nonlinear subdivision schemes. Applications in Image Processing. Found. Comput. Math., 193–225 (2006)

5. Amat, S., Donat, R., Liandrat, J., Trillo, J.C.: A fully adaptive multiresolution scheme for image processing. Math. Comput. Modelling 46, 2–11 (2007)
6. Amat, S., Liandrat, J.: On the stability of the PPH nonlinear multiresolution. Appl. Comput. Harmon. Anal. 18(2), 198–206 (2005)
7. Aràndiga, F., Donat, R.: Stability through synchronization in nonlinear multiscale transformations. SIAM J. Sci. Comput. 29(1), 265–289 (2007)
8. Aràndiga, F., Donat, R., Santagueda, M.: (in preparation)
9. Cavaretta, A.S., Dahmen, W., Micchelli, C.A.: Stationary subdivision. Mem. Am. Math. Soc. 93, 346–349 (1991)
10. Cohen, A., Dyn, N., Matei, B.: Quasilinear subdivision schemes with applications to ENO interpolation. Appl. Comput. Harmon. Anal. 15, 89–116 (2003)
11. Conti, C., Horman, K.: Polynomial reproduction for univariate subdivision schemes of any arity. J. Aprox. Theory 163, 413–437 (2011)
12. Dadourian K.: Schémas de Subdivision, Analyses Multirésolutions non-linéaires. Applications. PhD Thesis, Université de Provence, (2008), `http://www.cmi.univ-mrs.fr/~dadouria/these/rapport.pdf`
13. Dadourian, K., Liandrat, J.: Analysis of some bivariate non-linear interpolatory subdivision schemes. Numer. Algor. 48, 261–278 (2008)
14. Deslauriers, G., Dubuc, S.: Interpolation dyadique. In: Fractals, dimension non entiéres et application, Masson, Paris, pp. 44–45 (1987)
15. Dyn, N.: Subdivision schemes in computer aided geometric design, vol. 20, pp. 36–104. Oxford University Press, Oxford (1992)
16. Harten, A.: Multiresolution representation of data: A general framework. SINUM 33(3), 1205–1256 (1996)
17. Marquina, A.: Local piecewise hyperbolic reconstruction of numerical fluxes for nonlinear scalar conservation laws. SIAM J. Sci. Comput. 15(4), 892–915 (1994)
18. Marquina, A., Serna, S.: Power ENO methods: a fifth order accurate Weighted Power ENO method. J. Comput. Phys. 117, 632–658 (2004)

Tracking Level Set Representation Driven by a Stochastic Dynamics

Christophe Avenel[1], Etienne Mémin[2], and Patrick Pérez[3]

[1] Université Rennes 1
cavenel@inria.fr
[2] Inria
etienne.memin@inria.fr
[3] Technicolor Corporate Research
patrick.perez@technicolor.com

Abstract. We introduce a non-linear stochastic filtering technique to track the state of a free curve from image data. The approach we propose is implemented through a particle filter, which includes color measurements characterizing the target and the background respectively. We design a continuous-time dynamics that allows us to infer inter-frame deformations. The curve is defined by an implicit level-set representation and the stochastic dynamics is expressed on the level-set function. It takes the form of a stochastic partial differential equation with a Brownian motion of low dimension. Specific noise models lead to the traditional level set evolution law based on mean curvature motions, while other forms lead to new evolution laws with different smoothing behaviors. In these evolution models, we propose to combine local photometric information, some velocity induced by the curve displacement and an uncertainty modeling of the dynamics. The associated filter capabilities are demonstrated on various sequences with highly deformable objects.

Keywords: Tracking, Particle filtering, Level set, Continuous dynamic.

1 Introduction

The video tracking of an interface between two regions is a central process in numerous domains like medical imaging, meteorology or traffic control. Despite the many solutions proposed, no optimal solution exists yet for state variables defined on large dimensional spaces such as curves.

In order to deal with regions undergoing a complex deformation along time and potentially involving topological changes, we will confine ourself to a level set representation of the region boundaries. This representation has the great advantage to formulate the curve evolution within an Eulerian framework which avoids the definition of splitting/merging stratagem of Lagrangian splines curve representations when the curve is subject to topological changes.

Many tracking approaches proposed so far for the tracking of a level set curve representation are often defined as techniques implementing successive almost independent detection processes on each image of the image sequence [3,7,8,9,11,12]. Even if those techniques includes a temporal initialization strategy, they cannot truly be considered as tracking processes, as they do not guaranty any temporal coherence of the curve point trajectory. Such temporal incoherences are all the more pregnant that ambiguities due to clutter noise or illumination variations are observed. In addition to this, the few probabilistic techniques that have been proposed so far in the literature for curve tracking [5,14] are built upon *adhoc* linear models of the curve evolution law which limits them to the tracking of objects undergoing small or quasi-rigid deformations.

In the solution we propose, the evolution law is defined as a continuous-time stochastic dynamical model. The tracking is formulated as a stochastic filtering problem in which the available photometric data are filtered by such stochastic evolution laws. Stochastic filtering in high dimensional spaces is excruciatingly difficult to implement with particle or ensemble filters due to obvious sampling difficulties of high dimensional pdf., it is hence very important to devise dynamics that are the most accurate as possible. In the same time, we have to circumscribe the space of the random deformations applied on the curve in order to be able to efficiently draw samples, but also to make possible an efficient exploration of the considered curve's state space. To that end, the curve dynamics on which we rely is formulated as a stochastic transport equation defined directly on the implicit surface function. It includes constant displacement uncertainties along the curve tangent and normal directions. The transport velocity at each point of the curve is inferred from its past trajectory. This transport velocity field is computed through an additional vectorial level set function maintaining along time the correspondences between the current curve's points location and their original positions. The dynamics describing the evolution of this auxiliary function incorporates the uncertainties on the curve deformations as well. The curve dynamics is supplemented with a local photometric information that takes the form of a data-driven force, in order to guide more efficiently the curve prediction toward the next image observation.

2 Continuous-Time Dynamical Model of Level Set and Filtering

This section first recalls the general principles of particle filtering and presents the level set framework. Built on these ingredients, the proposed approach is then detailed.

2.1 Particle Filter

In this subsection, we introduce the filtering method we are using in the rest of the paper. The corresponding filters, called particle filters, are very general in the sense they enable coping with nonlinear dynamics and nonlinear measurement with additive eventually non-Gaussian noises. Denoting by $\mathbf{x}_{0:k}$ the trajectory from the initial time up to the current time instant k of the hidden Markov process we want to estimate from the whole set of past observations $\mathbf{z}_{1:k}$, a recursive expression of the filtering distribution $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ can be obtained from Bayes' law and the assumption that the measurements depends only on the current state:

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}) \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})}. \tag{1}$$

Particle filtering techniques implement an approximation of this density, using a sum of N Diracs centered on hypothesized locations in the state space. At each one of these locations - called particles - is assigned a weight $w_k^{(i)}$ describing his relevance. This approximation can be formulated as

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)} \delta_{\mathbf{x}_{0:k}}(\mathbf{x}_{0:k}). \tag{2}$$

It is impossible to simulate the samples directly from this unknown distribution. Particles are thus simulated from a proposal distribution $\pi(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$. This distribution, called the importance distribution, approximates the true filtering distribution. Each sample is then weighted using the ratio between the two distributions. The value of $w_k^{(i)}$ accounts for the deviation with respect to the unknown true distribution.

As a result, the target distribution will be fairly sampled by the particles $\mathbf{x}_{0:k}^{(i)}$ weighted by weights $w_k^{(i)}$, defined as

$$w_k^{(i)} = \frac{p(\mathbf{x}_{0:k}^{(i)}|\mathbf{z}_{1:k})}{\pi(\mathbf{x}_{0:k}^{(i)}|\mathbf{z}_{1:k})}. \tag{3}$$

To get the best efficiency the approximation needs obviously to be the closest as possible to the true distribution. However, any importance function can be chosen, with the only restriction that its support contains the target density's one. The importance ratio can be recursively computed assuming the importance density can be written in the following recursive form:

$$\pi(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = \pi(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})\pi(\mathbf{x}_k|\mathbf{z}_{1:k}, \mathbf{x}_{0:k-1}) \tag{4}$$

As $p(\mathbf{z}_k|\mathbf{z}_{1:k-1})$ is the same for every particle, it can be removed from relation (1), which leads to a general recursive update formulation of the weights at the current time when the measurement $\mathbf{z}_k$ becomes available:

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{z}_k|\mathbf{x}_k^{(i)})p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})}. \tag{5}$$

Using (2) and the normalized weights, it is then easy to obtain marginals of the complete filtering density:

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)} \delta_{\mathbf{x}_k}(\mathbf{x}_k). \tag{6}$$

Thus, by propagating the particles from time $k-1$ through the proposal density $\pi(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})$, and by weighting the sampled states with $w_k^{(i)}$, we obtain a sampling of the filtering law.

Asymptotically, for a number of particles tending to infinity, convergence toward the Bayesian filtering distribution of various classes of particle filters has been demonstrated [4] with a rate of $1/\sqrt{N}$. In practical implementations the number of particles is difficult to fix. The number of required particles to ensure the filter convergence depends on the state space dimension but also on the ability we have to draw samples in meaningful areas of this state space.

A resampling step of the particles is necessary to avoid the increase over time of the weight variance. Without this step, the number of significant particles decreases significantly along time. This procedure discards particles with weak weights, and duplicates particles with high weights.

When the proposal distribution is set to the dynamics, the weights updating rule (5) simplifies to the data likelihood $p(\mathbf{z}_k|\mathbf{x}_k^{(i)})$. This particular instance of the particle filter is called the *Bootstrap filter* and constitutes the filtering method we will use in this study.

2.2 Implicit Representation of the Curve

In order to cope with curve's deformation of any kinds, it is essential to rely on a curve representation enabling easily the handling of topological changes arising when the region of interest splits apart in several separated components or on the contrary reassembles. The Level set formalism [11,15] has been specifically introduced in that goal to bypass the deficiency of splines based curves representation to manage such situations. In this representation the curve $\mathcal{C}_t$ at time t is defined as the zero level set of a scalar function $\varphi(x,t) : \Omega \times \mathbb{R}^+ \rightarrow \mathbb{R}$:

$$\mathcal{C}_t = \{x \in \Omega | \varphi(x,t) = 0\}, \tag{7}$$

where Ω stands for the image spatial domain. The implicit surface function, φ, is chosen so as to have for instance positive values inside the curve and negative values outside. A common choice for the implicit function is the signed distance function but any other surface whose level set fits the curve of interest is possible. The surface evolution is then defined in order to stick at all time to the contour dynamics. This representation has the great advantage to allow describing through a single implicit surface a set of non-intersecting closed curves. The main geometric features of the curve that will be needed for the curve evolution computation can be directly obtained from the implicit surface. In particular, the inward unit normal and the mean curvature are respectively given by:

$$\boldsymbol{n} = \frac{\nabla\varphi}{\|\nabla\varphi\|} \text{ and } \kappa = \operatorname{div}\left(\frac{\nabla\varphi}{\|\nabla\varphi\|}\right) = \frac{1}{\|\nabla\varphi\|}(\Delta\varphi - \nabla\varphi^T\nabla^2\varphi\, \nabla\varphi), \tag{8}$$

where $\Delta\varphi$ and $\nabla^2\varphi$ denote the Laplacian and the Hessian of φ, respectively.

2.3 Stochastic Dynamics

Efficient random sampling in high dimensional state space are known to be problematic. Some work have been done on Quasi-Random Sampling-High Dimensional Model Representation [6], but there is no optimal solution for random sampling in our case. In order to circumvent this problem, we will restrain the potential uncertainty on the curve's deformations to belong to a space of low dimension. To that end, the uncertainty on the curve deformation will be defined from two independent constant Brownian motions directed along the curve's normal and tangent:

$$d\mathcal{C}_t = v_n \boldsymbol{n} dt + \sigma_n \boldsymbol{n} dB_{n,t} + \sigma_\tau \boldsymbol{n}^\perp dB_{\tau,t}. \tag{9}$$

In this equation $dB_{n,t}$ and $dB_{\tau,t}$ denote the two Brownian motions, σ_n and σ_τ are two diffusion coefficients, $\boldsymbol{n}$ is the unit vector normal to the curve and $v_n = \mathbf{v}^T \boldsymbol{n}$ is the projection on the curve's normal of a deterministic transport velocity field $\mathbf{v}$. The random Brownian diffusion terms encode the curve deformation uncertainty. The deterministic transportation drift component will be further detailed in Section 2.4.

The surface φ can be used to express the deformation of the curve (9) on space Ω:

$$d\mathcal{X}_t = w_n^* \frac{\nabla\varphi}{|\nabla\varphi|} dt + \sigma_n \frac{\nabla\varphi}{|\nabla\varphi|} dB_{n,t} + \sigma_\tau \frac{\nabla\varphi}{|\nabla\varphi|}^\perp dB_{\tau,t}. \tag{10}$$

where w_n^* is an extension on the whole image domain of the curve's drift strength component.

The curve at time t is defined by construction through its implicit representation at time t:

$$\varphi(.,t) = \varphi(.,0) + \int_0^t d\varphi(.,s). \tag{11}$$

It is thus a function of the stochastic process $\mathcal{C}_t$.

As for a fixed point x, $\varphi(x,t)$ is a semi-martingale, the differential of $\varphi(\mathcal{X}_\sqcup, \sqcup)$ has to be calculated through the Ito-Wentzell formula (differential of the composition of two stochastic process):

$$d\varphi(x,t) = d\varphi_t(x) + \nabla\varphi^T d\mathcal{X} + \frac{1}{2}\sum_{i,j} d\left\langle \mathcal{X}_t^i, \mathcal{X}_t^j \right\rangle \frac{\partial^2\varphi}{\partial x_i \partial x_j} + \sum_i d\left\langle \frac{\partial\varphi}{\partial x_i}, \mathcal{X}_t^i \right\rangle. \tag{12}$$

It then leads to

$$\begin{aligned} d\varphi_t(x) &= b(y,t)dt + f(y,t)dB_{n,t} \\ &= -\nabla\varphi^T w_n^* dt - \frac{\sigma_\tau^2 dt}{2}\left(\Delta\varphi - \frac{1}{|\nabla\varphi|^2}\nabla\varphi^T \Delta\varphi \nabla\varphi\right) \\ &+ \frac{\sigma_n^2 dt}{2}\left(\frac{1}{|\nabla\varphi|^2}\nabla\varphi^T \Delta\varphi\nabla\varphi\right) - \sigma_n |\nabla\varphi| dB_{n,t}. \end{aligned} \tag{13}$$

Compared to the classical deterministic level set differential, this expression introduces a Brownian component directed along the curve normal and an additional second-order smoothing term. The mean curvature component results from the introduction of the curve motion uncertainty along the curve tangent. It is worth noting that this formulation permits to interpret the mean curvature motion component as a consequence of the uncertainty one has on the curve's deformation along its tangent. This stochastic representation of the curve temporal evolution will enable us to draw samples of forecasted deformed curves.

2.4 Velocity Computation by Keeping Curve's Point Correspondences

The evolution law introduced in the previous section depends on a transport velocity field. This transport component could be derived from motion measurements estimated from the image sequence. However this solution has several drawbacks. It increases substantially the computational cost of the approach and requires the use of an external estimation technique. Furthermore, such a solution is not adapted to handle occlusions areas, where motion estimation is prone to errors. The use of such velocity measures would require thus the introduction of an additional occlusion detection mechanism and the definition of an alternative velocity field when occlusions occur. We propose instead to infer directly the velocity from each particle displacements, via a second implicit representation that keeps track of each point's starting location in the image plane. As proposed in [13] to keep this point correspondences, we introduce an additional vectorial level set $\boldsymbol{w}$ encoding on the level set domain the transportation of the curve's point location at the initial time. Keeping track of these backward point correspondences between the current evolving curve and a recent predecessor will allow us to derive an estimate of the curve's point velocity field.

More precisely, introducing the previous Cartesian coordinates of the curve's points location and encoding them through a vector-valued level set function $\boldsymbol{\psi}^k : \mathbb{R}^2 \times \mathbb{R}^+ \to \mathbb{R}^2$ such that $\boldsymbol{\psi}^k(x,t)$ define the location that point $x \in \Omega$ at time $t \in [k, k-1]$ was occupying at previous instant $k-1$:

$$\boldsymbol{\psi}^k(x, k-1) = x. \tag{14}$$

This new level set function is intrinsically attached to the curve and undergoes deformations dictated by the stochastic curve's evolution law (9). Applying in the same way as previously the Ito formula, its differential reads:

$$\begin{aligned} d\psi^i(x,t) &= d\psi^i_t(x) + (\nabla\psi^i_t)^T d\mathcal{X}_{\sqcup} \\ &\quad + \frac{1}{2}\sum_{i,j} d\left\langle \mathcal{X}^i_t, \mathcal{X}^j_t \right\rangle \frac{\partial^2 \psi^i_t}{\partial x_i \partial x_j} + \sum_i d\left\langle \frac{\partial \psi^i_t}{\partial x_i}, \mathcal{X}^i_t \right\rangle = 0. \end{aligned} \tag{15}$$

In this case, the tangential component of the Brownian motion is not null. This function enables us to define the curve's transportation component (13) as

$$\mathbf{v}(x,t)dt = \frac{1}{\Delta t}(x - E(\boldsymbol{\psi}^{k-1}(x, k-1)|\mathcal{C}_{k-1})),\ t \in [k-1, k], \tag{16}$$

where $E(\cdot|\mathcal{C}_t)$ denotes the expectation with respect to the the path of $\mathcal{C}_t$ up to time t (formally the natural filtration associated to the process $\mathcal{C}_t$). This transportation velocity field is thus defined as a deterministic function computed from the realization of the curve (through its level set function φ_t) up to the previous instant $k-1$. Considering the particle approximation this velocity field is computed as

$$\mathbf{v}(x,t)dt = \frac{1}{\Delta t}\left(x - \frac{1}{N}\sum_{i=1}^{N} w^{(i)}\boldsymbol{\psi}^{k-1,(i)}(x,k-1)\right),\ t \in [k-1,k], \tag{17}$$

where $\boldsymbol{\psi}^{k-1,(i)}$ corresponds to the auxiliary level set distance function associated to particle $\varphi^{(i)}$ and $w^{(i)}$ its importance weight.

Let us note that as $\mathbf{v}(x,t)$ can only be used from instant $t=1$, we have to define a transport component for t between 0 and 1. This initial transportation component is set to the velocity field estimated from the two first images through an optical-flow estimator.

The velocity field is complemented with a local potential $F(\varphi)$ that corresponds to the Chan and Vese segmentation operator [2] extended to color histograms. It allows us to refine the tracking by taking into account color information, using local measurements that are inexpensive to compute[1].

The two components are combined linearly with proportions $\beta(t) \in [0,1]$ and $1-\beta(t)$ respectively, yielding:

$$v_n = \beta(t)\mathbf{v}^T\boldsymbol{n} + (1-\beta(t))\partial_\varphi F(\varphi). \tag{18}$$

For our tracking purpose, the photometric component is especially helpful in the temporal vicinity of the second image, whereas the velocity component is more likely to be meaningful in the temporal vicinity of the first image. As a consequence we choose to change gradually the proportion of each components according to

$$\beta(t) = t - k + 1, t \in [k-1,k]. \tag{19}$$

Equations (13-15) allows us to forecast instances of deformed curves and to sample the proposal distribution. The importance weights of these curve particles have to be updated from the data likelihood. We detail in the following section this likelihood and propose a technique for estimating the variances of the curve evolution law uncertainties.

3 Measurement Models and Parameters Estimation

3.1 Likelihood Definition

In bootstrap filters, the data likelihood associated to each particle directly determines its weight. It is therefore crucial for this distribution to be sufficiently

[1] As a consequence of the use of a data-driven force in the dynamics, the state space model is not anymore a classical hidden Markov model. It has been shown, however, that standard derivations that lead to filtering recursion can still be conducted with such models, leading to so-called conditional filters [1].

discriminant in order to discard curves which are too distant from the intended result. To this end, we choose to define a likelihood that depends on the similarity between the color distributions inside the curve at times $t = 0$ and $t = k$ respectively. This is a classical choice in literature [10]. For each particle, it reads:

$$p(\mathbf{z}_t | x_t^{(i)}) \propto \exp^{-\lambda d(h_0, h_k^{(i)})}, \tag{20}$$

where d is the Bhattacharyya distance between h_0 the reference interior color histogram instantiated at time 0 and $h_k^{(i)}$ the interior color histogram associated to the i-th level set sample at time k, and λ is a positive parameter. For discrete probability distributions p and q defined over the same domain X, the Bhattacharyya distance is defined as

$$d(p,q) = \left(1 - \sum_{x \in X} \sqrt{p(x)q(x)}\right)^{1/2}. \tag{21}$$

3.2 Parameters Estimation

The dynamics defined through equations (13-15) involves two diffusion coefficient related to the uncertainty associated to the curve motion. These parameters σ_n and σ_τ can be derived from the displacements field between two consecutive images, as follows. Let $\mathbf{u}(x,t)$ be the displacement of point $x \in \Omega$ at time k to its corresponding position at time $k+1$, according to the evolution of the implicit function φ conditioned on the past observation. We assume that this displacement field is a noisy version of $\mathbf{v}$,

$$\mathbf{u}(x,k) = \mathbf{v}(x,k)dt + \sigma_n \boldsymbol{n} dB_{n,t} + \sigma_\tau \boldsymbol{n}^\perp dB_{\tau,t}, \tag{22}$$

with noises along the normal and the tangent of φ level-lines having same characteristics as those in (9). We are therefore making here the assumption that the noises associated with the level set displacement and the curve noises are collinear and have the same variances. Furthermore, we assume that the transport velocity field is such that $\mathbf{v}(x,t) = \mathbb{E}(\mathbf{u}(x,k))/dt$. The empirical covariances with respect to the filtering law of this observed displacement along the curve normal and tangent provide an estimation of the noise variances σ_n^2 and σ_τ^2:

$$\sigma_n^2 = \frac{1}{N-1} \sum_{i=1:N} \left(\frac{1}{n-1} \sum_{x \in \mathcal{C}^{(i)}} w^{(i)} \left((\mathbf{u}(x,k) - \mathbf{v}(x,k)) \cdot \boldsymbol{n}(x,t) \right)^2 \right) \tag{23}$$

$$\sigma_\tau^2 = \frac{1}{N-1} \sum_{i=1:N} \left(\frac{1}{n-1} \sum_{x \in \mathcal{C}^{(i)}} w^{(i)} \left((\mathbf{u}(x,k) - \mathbf{v}(x,k)) \cdot \boldsymbol{n}^\perp(x,t) \right)^2 \right). \tag{24}$$

For the time interval between the two first images, the values of these parameters are computed from the initial motion field used to initialize our filter. The next section shows results obtained for different kinds of image sequences.

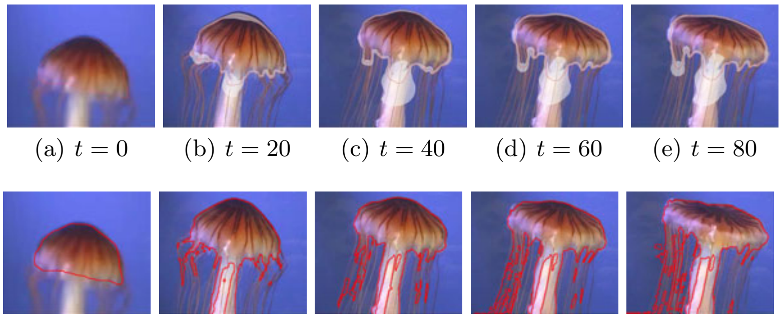
4 Experiments and Results

This section reports several tracking results obtained with our approach. We aim here at highlighting the main abilities of the method. Note that in all these experiments, the number of particles is fixed as $N = 100$. Each curve initialization is performed manually, at the initial time. This initial curve will be systematically plotted for all the sequences tested.

4.1 Interest of a Continuous-Time Stochastic Dynamics and Auxiliary Level-Set

One of the main distinctive features of our approach is that it relies on a continuous-time stochastic dynamics. This allows the exploitation of temporal continuity even when deformations between successive images are drastic, as illustrated in the jellyfish sequence in Fig. 1 (a - e). This is in contrast with approaches relying on a succession of segmentation tasks. On the same sequence, for instance, the Chan-Vese segmentation method fails to recover an appropriate tracking of the delineated region (Fig. 1 : f - j). In this approach the segmentation process is initialized with the results obtained on the previous frame. There is no explicit handling of the possible deformation between two images. On the tracking results, in addition to the mean curve we plot through a white band a representation of the variance of the filtering. This aspect is further detailed in the next section.

Getting inter-frame tracking information is also of potential interest in context where successive images of the sequence are fairly distant in time, e.g., in meteorology and weather forecast.


(a) $t = 0$ (b) $t = 20$ (c) $t = 40$ (d) $t = 60$ (e) $t = 80$

(f) $t = 0$ (g) $t = 20$ (h) $t = 40$ (i) $t = 40$ (j) $t = 40$

Fig. 1. Tracking of a jellyfish, with a highly deformable body using our method (a - e) and deterministic Chan-Vese technique (f - j)

4.2 Variance Visualization and Analysis

Beyond the tracking results provided by the weighted mean curve, local confidence assessment via local variance visualization (or analysis) is an interesting feature of our approach. The weighted set of implicit function samples provided by particle filtering permits such a visualization. The weighted local variance of the level set functions around the mean level set, which is obtained by computing $V = \sum_{i=1}^{N} w^{(i)}(\varphi^{(i)} - \overline{\varphi})^2$, is here represented by a white band around the mean curve, the lower the variance, the narrower the uncertainty band. To illustrate this variance representation, we show results obtained on a second sequence (Fig. 2) showing a tiger running. In this sequence the colors of the background and the target are very close, which is an important source of ambiguities. We can observe in particular that for areas around the legs, the uncertainty is important. We observe that the results are good without using any external estimated motion field.

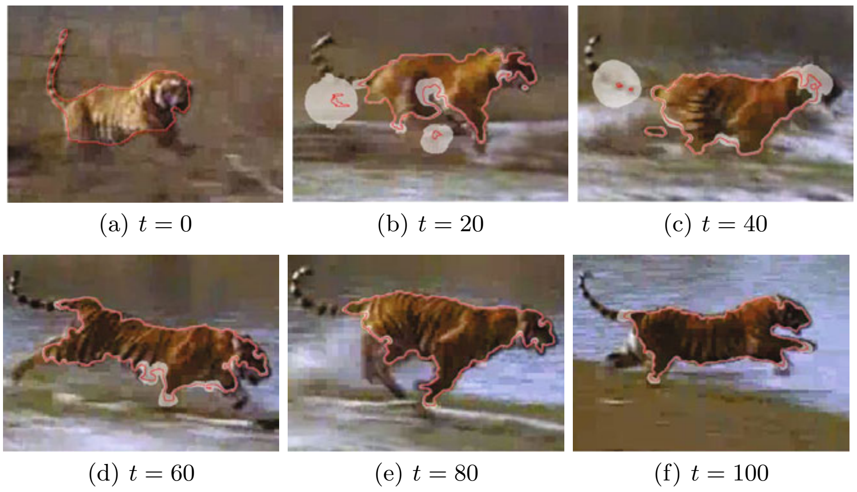
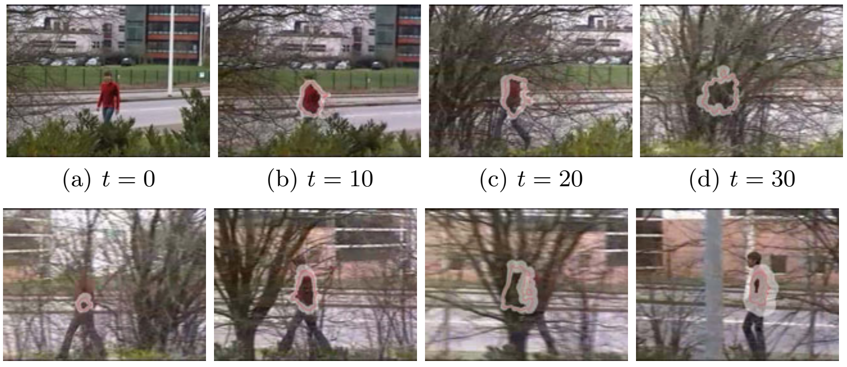

(a) $t = 0$ (b) $t = 20$ (c) $t = 40$ (d) $t = 60$ (e) $t = 80$ (f) $t = 100$

Fig. 2. Tracking of running tiger with our particle filter on the space of implicit functions

To be able to access to an estimation of the tracking uncertainty is a great advantage of our technique. This should be of great interest in several application domains such as medical imaging in which the ability to quantify locally the quality of a result is essential for end-users.

4.3 Occlusions Management

One of the advantage provided by our transport velocity formulation compared to any optical flow measurements, is that it authorizes a natural handling of occlusion situations. As a matter of fact, no matter the region of interest be visible or not a velocity measure of the curve's is always available. This measure can thus be used at all instants without any distinction on the visibility or not of the considered point. There is here no need of empirical external occlusion detectors. For example, on Fig. 3, the person disappears on frames d and g, but


(a) $t = 0$ (b) $t = 10$ (c) $t = 20$ (d) $t = 30$

(e) $t = 40$ (f) $t = 50$ (g) $t = 60$ (h) $t = 70$

Fig. 3. Example of occlusions on a body walking behind trees

tracking relocks on it after the occlusions end. We can observe that during the occlusions, the uncertainty is growing all around the curve, as the parameter σ_n is growing.

As for the computational load. This approach can be straightforwardly on a multicore or grid computer as all the particles are independent. Only the weighting and the resampling requires communications between the processors. All these results have been run on a grid composed of 100 nodes, and the approach took nearly 5 minutes for a 100 images sequences. Let us note however that the level set has been implemented with a narrow band efficient implementation and could be hence much more faster than the present version.

5 Conclusion

In this paper we have proposed a probabilistic filtering method for the tracking of level sets. The underlying model combines discrete-time image measurements with a continuous-time stochastic dynamics. This dynamics relies on two different uncertainties on the curve motion, directed respectively along the curve normal and along the curve tangent. It also includes a transport vector field that combines an image-based force (related to local photometry) and a velocity induced by previous curve displacements and deformations. The measurement considered in this model are built from color histograms of the object delineated by the user at the initial time. The implementation of the filter is done via a particle filter whose proposal density amounts to simulating several steps of a discretized stochastic differential equation.

We have illustrated on several examples the interest of the continuous-time dynamics and of the estimation confidence assessment that proper stochastic filtering permits via the approximation of the filtering distribution. In particular, displaying the estimation uncertainty along the curve, which is done by computing the variance of each point of this curve, could be an interesting tool for, e.g., biology or meteorology imaging. Also, the ability to show inter-frame results is an other potential advantage that could be used to infer potential curve deformations between the two frames instant.

Finally, besides allowing to deal with occlusions, using the velocity of the curve could help predicting the evolution of this curve for a few frames ahead of time, which should be useful in various domains for forecast application.

References

1. Arnaud, E., Mémin, E.: Partial linear gaussian model for tracking in image sequences using sequential monte carlo methods. IJCV 74(1), 75–102 (2007)
2. Chan, T., Vese, L.: An active contour model without edges. In: Scale-Space Theories in Computer Vision, pp. 141–151 (1999)
3. Cremers, D., Soatto, S.: Motion competition: A variational framework for piecewise parametric motion segmentation. IJCV 62(3), 249–265 (2005)
4. Crisan, D., Doucet, A.: A survey of convergence results on particle filtering methods for practitioners. IEEE Transactions on Signal Processing 50(3), 736–746 (2002)
5. Dambreville, S., Rathi, Y., Tannenbaum, A.: Tracking deformable objects with unscented kalman filtering and geometric active contours. In: American Control Conference, pp. 1–6 (2006)
6. Feil, B., Kucherenko, S., Shah, N.: Comparison of monte carlo and quasi monte carlo sampling methods in high dimensional model representation. In: Proceedings of the 2009 First International Conference on Advances in System Simulation, pp. 12–17. IEEE Computer Society, Washington, DC, USA (2009)
7. Goldenberg, R., Kimmel, R., Rivlin, E., Rudzsky, M.: Fast geodesic active contours. IEEE Trans. on Image Processing 10(10), 1467–1475 (2001)
8. Kimmel, R., Bruckstein, A.M.: Tracking level sets by level sets: a method for solving the shape from shading problem. Comput. Vis. Image Underst. (1995)
9. Niethammer, M., Tannenbaum, A.: Dynamic geodesic snakes for visual tracking. In: CVPR (1), pp. 660–667 (2004)
10. Nummiaro, K., Koller-Meier, E., Gool, L.V.: An adaptive color-based particle filter. Image and Vision Computing 21(1), 99–110 (2003)
11. Osher, S., Sethian, J.A.: Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. Journal of Computational Physics 79, 12–49 (1988)
12. Paragios, N., Deriche, R.: Geodesic active regions: a new framework to deal with frame partition problems in computer vision. J. of Visual Communication and Image Representation 13, 249–268 (2002)
13. Pons, J.P., Hermosillo, G., Keriven, R., Faugeras, O.: Maintaining the point correspondence in the level set framework. Journal of Computational Physics 220(1), 339–354 (2006)
14. Rathi, Y., Vaswani, N., Tannenbaum, A., Yezzi, A.: Tracking deforming objects using particle filtering for geometric active contours. IEEE Trans. Pattern Analysis and Machine Intelligence 29(8), 1470–1475 (2007)
15. Sethian, J.A.: Theory, algorithms, and applications of level set methods for propagating interfaces. Acta Numerica 5(-1), 309–395 (1996)

G^2 Hermite Interpolation with Curves Represented by Multi-valued Trigonometric Support Functions

Bohumír Bastl, Miroslav Lávička, and Zbyněk Šír

University of West Bohemia, Faculty of Applied Sciences, Department of Mathematics,
Univerzitní 8, 301 00 Plzeň, Czech Republic
{bastl,lavicka,zsir}@kma.zcu.cz

Abstract. It was recently proved in [27] that all rational hypocycloids and epicycloids are Pythagorean hodograph curves, i.e., rational curves with rational offsets. In this paper, we extend the discussion to a more general class of curves represented by trigonometric polynomial support functions. We show that these curves are offsets to translated convolutions of scaled and rotated hypocycloids and epicycloids. Using this result, we formulate a new and very simple G^2 Hermite interpolation algorithm based on solving a small system of linear equations. The efficiency of the designed method is then presented on several examples. In particular, we show how to approximate general trochoids, which, as we prove, are not Pythagorean hodograph curves in general.

1 Introduction

The planar G^2 Hermite interpolation problem requires a construction of a suitable planar curve matching a given set of data, which consists of pairs of points, unit tangent vectors and signed curvatures at those points. Then, a solution of this interpolation problem allows one to construct a G^2 Hermite interpolating spline between neighbouring pairs of points, cf. [5,9,12,21,22,32] for some algorithms and related discussions. This property is desirable in CAD and CAGD applications – e.g. for planning cutting paths of CNC machines, for computing robot trajectories, in the design of highways or railways.

Pythagorean hodograph (PH) curves are rational curves distinguished by the property that their offsets are also rational, which is another attractive property desirable in CAD and CAGD applications, for more details see [8] and references therein. In addition, the Pythagorean hodograph property under Minkowski metric was studied e.g. in [4,23,15,16,17,18]. However, there are not many algorithms in curve design with segments of Pythagorean hodograph curves preserving G^2 continuity – see e.g. [11,14,31,13,2,6,24]. Moreover, these techniques are often based on solving difficult systems of nonlinear equations with a tough and time-consuming subsequent discussion.

It was shown recently in [27] that all rational hypocycloids and epicycloids (cf. [7,33]) are rational PH curves. In addition, a simple G^1 Hermite interpolation algorithm using hypo/epicycloids with rational offsets was designed. That algorithm is based on an application of the so called support function representation of algebraic curves, which was introduced to geometric modelling in [10,28,29] and later used e.g. in [1,20]. In this paper, we extend this discussion to a general class of curves represented by trigonometric polynomial support functions, which are nothing else than offsets to translated convolutions of scaled and rotated rational hypocycloids and epicycloids. The reader who is more interested in the theory of convolutions of algebraic curves can find more details e.g. in [29,19,20,30].

We will prove that all curves supported by trigonometric polynomials (in generalized sense) are PH and show that they are very suitable for constructing G^2 Hermite interpolating splines with rational offsets. The main advantage of the designed algorithm, compared to other G^2 Hermite interpolation techniques, is its simplicity as our method is based only on solving a simple system of (generally 6) linear equations. Nevertheless, since hypo/epicycloids (nor their convolutions) do not contain inflection points, a constructed spline possesses the same property. Hence, if the data points fail to have the same signs of curvatures, a simple preprocessing step is needed, i.e., we have to insert a data point at the inflection. This reduces the G^2 to G^1 continuity at that point.

The remainder of the paper is organized as follows. Section 2 recalls some basic facts concerning support functions and fundamental theory of hypo/epicycloids. Section 3 is devoted to the curves supported by trigonometric polynomials and their relation to hypo/epicycloids. In addition, we provide a method for computing their rational PH parameterizations. In Section 4, we formulate an algorithm for G^2 Hermite interpolation with arcs of curves represented by trigonometric polynomial support functions. The designed algorithm is then demonstrated on several examples and approximation order is studied. Finally, we conclude the paper in Section 5.

2 Preliminaries

We recall some fundamental properties of hypo/epicycloids, and summarize the basic theory of support function representation of algebraic curves in connection with their convolutions and with the theory of rational offsets.

2.1 Support Function Representation of Algebraic Curves

Any algebraic curve in the plane (which is not a line) has the *dual representation*

$$D(\mathbf{n}, h) = 0, \tag{1}$$

where D is a homogeneous polynomial in h and $\mathbf{n} = (n_1, n_2)^\top$ (the normal vector). This equation specifies the set of tangents of the curve. In addition, if $\mathbf{n}$ is unit then h expresses the oriented distance of the corresponding tangent to the origin.

If the partial derivative $\partial D/\partial h$ does not vanish at $(\mathbf{n}_0, h_0) \in \mathbb{R}^3$ and $D(\mathbf{n}_0, h_0) = 0$ holds, then (1) implicitly defines a function

$$\mathbf{n} \mapsto h(\mathbf{n}) \tag{2}$$

in a certain neighborhood of $(\mathbf{n}_0, h_0) \in \mathbb{R}^3$. The restriction of this function to the unit circle $S_1 = \{\mathbf{n} \in \mathbb{R}^2 : |\mathbf{n}| = 1\}$ is the so called (generally multi-valued) *support function* of the curve.

On the other hand, from any smooth real function on S_1 we can reconstruct the corresponding curve by the mapping $\mathbf{x}_h : S_1 \to \mathbb{R}^2$

$$\mathbf{x}_h(\mathbf{n}) = h(\mathbf{n})\mathbf{n} + h'(\mathbf{n})\mathbf{n}^{\perp}, \tag{3}$$

where $\mathbf{n}^{\perp}$ denotes the clockwise rotation of $\mathbf{n}$ about the origin by the angle $\frac{\pi}{2}$ and h' is the derivative with respect to the arc-length, see [28].

For later use we recall how the support function $h(\mathbf{n})$ is affected by selected geometric operations, cf. [25,28]:

$$\begin{array}{rl} \text{(i)} & \textit{translation}: h(\mathbf{n}) \mapsto h(\mathbf{n}) + \mathbf{v}\cdot\mathbf{n}, \text{ where } \mathbf{v} \in \mathbb{R}^2 \text{ is a translation vector;} \\ \text{(ii)} & \textit{rotation}: h(\mathbf{n}) \mapsto h(\mathbf{A}\mathbf{n}), \text{ where } \mathbf{A} \text{ is an orthogonal matrix from } SO(2); \\ \text{(iii)} & \textit{scaling}: h(\mathbf{n}) \mapsto \lambda h(\mathbf{n}), \text{ where } \lambda \in \mathbb{R} \text{ is a scaling factor;} \\ \text{(iv)} & \textit{offsetting}: h(\mathbf{n}) \mapsto h(\mathbf{n}) + \delta, \text{ where } \delta \in \mathbb{R} \text{ is an offsetting distance.} \end{array} \tag{4}$$

Moreover, the support function representation is very suitable for describing the *convolution* $\mathcal{C}_3 = \mathcal{C}_1 \star \mathcal{C}_2$ of curves $\mathcal{C}_1, \mathcal{C}_2$ as this operation corresponds to the sum of the associated support functions

$$h_3 = h_1 + h_2, \tag{5}$$

see [28,20] for more details.

Another very useful property of the support function representation (especially in connection with G^2 Hermite interpolation problem) is that it can be efficiently used for describing the *signed curvature* of a given curve, cf. [28], in the form

$$\kappa(\theta) = -\frac{1}{h(\theta) + h''(\theta)}. \tag{6}$$

2.2 Rational Hypocycloids and Epicycloids

A *hypocycloid* is a plane curve generated by the trace of a fixed point $\mathbf{x}$ on a circle with radius r which rolls without sliding within a fixed circle with radius $R > r$. If the fixed circle is centered at the origin and $\mathbf{x}(0) = (0, R-r)^{\top}$, then the hypocycloid is parameterized by

$$\mathbf{x}(\varphi) = (R-r)\cdot\mathbf{n}(\varphi) - r\cdot\mathbf{n}(k\varphi), \quad k = 1 - \frac{R}{r}, \tag{7}$$

where $\mathbf{n}(\varphi) = (\sin\varphi, \cos\varphi)^{\top}$.

If the moving circle roles outside the fixed circle, we generate an *epicycloid*, whose parametric equations are obtained by using a negative r in (7). Moreover, the beautiful *double generation theorem* (first noticed by Daniel Bernoulli in 1725) guarantees that, without loss of generality, we can assume $R > 2r$ in the rest of this paper (see [27,33]).

Let us recall that hypo/epicycloids are rational curves if and only if k is a rational number, otherwise they are transcendental. In what follows we assume that $R : r$ is a rational number and we introduce the name *HE-cycloid* as a unified term for both rational hypo/epicycloids.

It was proved in [27] than any HE-cycloid (7) admits the support function representation in the form

$$h(\theta) = (R - 2r)\cos\left(\frac{R}{R-2r}\theta\right) \tag{8}$$

with respect to the parameterization $\mathbf{n}(\theta) = (\sin\theta, \cos\theta)^\top$. The coordinates are chosen such that the fixed circle is centered at the origin and for $\theta = 0$ the center of the rolling circle is located at $(0, R-r)^\top$ and the tracing point starts at $(0, R-2r)^\top$.

If we set $\varrho = \mathrm{GCD}(R, r)$, $a = R/\varrho$ and $b = (R-2r)/\varrho$ then (8) can be rewritten as follows

$$h(\theta) = (b\varrho)\cos\left(\frac{a}{b}\theta\right). \tag{9}$$

As the multiplication by a constant factor $b\varrho$ represents scaling, cf. (4-iii), we omit this factor and introduce the so called *canonical HE-cycloids with parameters* a, b described by

$$h(\theta) = \cos\left(\frac{a}{b}\theta\right) \tag{10}$$

and denoted by $\mathcal{C}^a_b$. Obviously, (10) represents an epicycloid for $a < b$ and a hypocycloid for $a > b$. For more details about geometric meaning of a, b see [27]. Many famous rational curves are examples of HE-cycloids – e.g. $\mathcal{C}^1_2$ is the *nephroid*, $\mathcal{C}^1_3$ is the *cardioid*, $\mathcal{C}^2_1$ is the *astroid* and $\mathcal{C}^3_1$ is the *deltoid*, cf. Fig. 1.

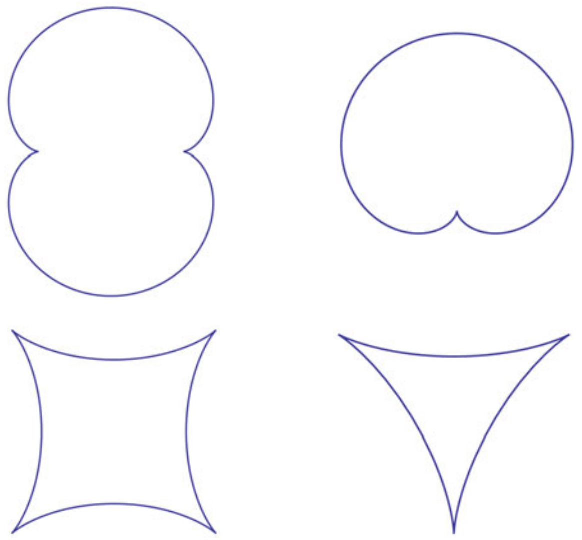


Fig. 1. Examples of HE-cycloids – the nephroid $\mathcal{C}^1_2$, the cardioid $\mathcal{C}^1_3$, the astroid $\mathcal{C}^2_1$ and the deltoid $\mathcal{C}^3_1$

Furthermore, let us recall that the support function representation was used in [27] to prove that all HE-cycloids are rational Pythagorean hodograph (PH) curves, i.e., they are rational curves with rational offsets. The reader who is more interested in the theory of PH curves is referred to [8] and references therein.

3 Rational Curves Supported by Trigonometric Polynomials

In this section, we will study curves whose support functions can be expressed as linear combinations of the support functions of the canonical HE-cycloids.

3.1 Trigonometric Polynomials and Convolutions of HE-Cycloids

Let us consider all curves supported by the trigonometric function

$$h(\theta) = A_0 + \sum_{q=1}^{\ell}\sum_{p=1}^{n_q}\left(A_{pq}\cos\left(\frac{p}{q}\theta\right) + B_{pq}\sin\left(\frac{p}{q}\theta\right)\right), \tag{11}$$

where $A_{pq}, B_{pq} \in \mathbb{R}$, $p, q \in \mathbb{N}$. By setting $\psi = \theta/\mathrm{LCM}(2,\ldots,\ell)$, where LCM denotes the least common multiple, (11) can be rewritten into a standard *trigonometric polynomial* defined by

$$A_0 + \sum_{m=1}^{N} A_m\cos(m\psi) + \sum_{m=1}^{N} B_m\sin(m\psi), \tag{12}$$

where $A_m, B_m \in \mathbb{R}$, $m \in \mathbb{N}$. In what follows, we will work with the notion trigonometric polynomial in this broader sense. Trigonometric polynomials are widely used e.g. for trigonometric approximations of periodic functions.

Now, we will study a relation of the functions (11) to the canonical HE-cycloids. One can see that A_0 describes offsetting, cf. (4-iv), and the term

$$(B_{11} + B_{22} + \ldots)\sin\theta + (A_{11} + A_{22} + \ldots)\cos\theta = v_1n_1 + v_2n_2 \tag{13}$$

represents a translation, see (4-i). Next, we consider terms of the form

$$A_{pq}\cos\left(\frac{p}{q}\theta\right) + B_{pq}\sin\left(\frac{p}{q}\theta\right), \quad q \neq p. \tag{14}$$

$\mathcal{C}_q^p$ scaled through a factor λ and rotated through an angle α_q, cf. (4-ii,iii), has the support function

$$\lambda\cos\left[\frac{p}{q}(\theta-\alpha_q)\right] = \underbrace{\lambda\cos\left(\frac{p}{q}\alpha_q\right)}_{A_{pq}}\cos\left(\frac{p}{q}\theta\right) + \underbrace{\lambda\sin\left(\frac{p}{q}\alpha_q\right)}_{B_{pq}}\sin\left(\frac{p}{q}\theta\right). \tag{15}$$

Inversely, from A_{pq}, B_{pq} we can easily compute λ and α_q. Finally, sums of the terms (14) represent the convolutions of associated primal curves, cf. (5). An example of the convolution of two HE-cycloids is illustrated in Fig. 2. Hence, we may formulate

Proposition 1. *The curve given by the support function* (11) *is an offset to a translated convolution of scaled and rotated canonical HE-cycloids.*

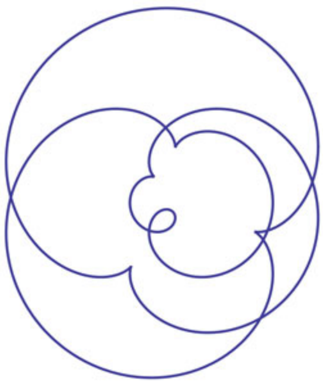


Fig. 2. Convolution of the nephroid $\mathcal{C}_2^1$ and the cardioid $\mathcal{C}_3^1$

3.2 Rational Offsets of Convolutions of HE-Cycloids

We will study curves supported by the function (14) from the point of view of the rationality of their offsets. We will look for a simultaneous rational parameterization of h and of $\mathbf{n} \in S_1$, which guarantees that the associated primal curve is a PH curve.

Proposition 2. *The convolution of two scaled and rotated canonical HE-cycloids is a rational Pythagorean hodograph curve and thus it possesses rational offsets.*

Proof. We consider two scaled and rotated canonical HE-cycloids supported by the functions

$$h_1 = A_1 \cos\left(\frac{a_1}{b_1}\theta\right) + B_1 \sin\left(\frac{a_1}{b_1}\theta\right), \quad h_2 = A_2 \cos\left(\frac{a_2}{b_2}\theta\right) + B_2 \sin\left(\frac{a_2}{b_2}\theta\right) \tag{16}$$

and their convolution represented by (5). Firstly, we convert the both arguments to a common denominator $b = \mathrm{LCM}(b_1, b_2)$ and work with $(\bar{a}_1\theta)/b$ and $(\bar{a}_2\theta)/b$

Using the de Moivre's formula and the binomial theorem we get the following equivalent expressions

$$\left[\cos\left(\frac{p}{q}\theta\right) + i \sin\left(\frac{p}{q}\theta\right)\right]^q = \begin{cases} \cos(p\theta) + i\sin(p\theta), \\ \sum_{j=0}^{q} \binom{q}{j} \left[\cos\left(\frac{p}{q}\theta\right)\right]^{q-j} \left[i \sin\left(\frac{p}{q}\theta\right)\right]^j . \end{cases} \tag{17}$$

Comparing the real and imaginary parts, respectively, we obtain the formulae

$$\cos(p\theta) = \sum_{j=0}^{\lfloor \frac{q}{2} \rfloor} (-1)^j \binom{q}{2j} \left[\cos\left(\frac{p}{q}\theta\right)\right]^{q-2j} \left[\sin\left(\frac{p}{q}\theta\right)\right]^{2j}, \tag{18}$$

$$\sin(p\theta) = \sum_{j=0}^{\lfloor \frac{q-1}{2} \rfloor} (-1)^j \binom{q}{2j+1} \left[\cos\left(\frac{p}{q}\theta\right)\right]^{q-2j-1} \left[\sin\left(\frac{p}{q}\theta\right)\right]^{2j+1} . \tag{19}$$

It is also well known that

$$\cos(p\theta) = \sum_{j=0}^{\lfloor \frac{p}{2} \rfloor} (-1)^j \binom{p}{2j} (\cos\theta)^{p-2j} (\sin\theta)^{2j}\,, \tag{20}$$

$$\sin(p\theta) = \sum_{j=0}^{\lfloor \frac{p-1}{2} \rfloor} (-1)^j \binom{p}{2j+1} (\cos\theta)^{p-2j-1} (\sin\theta)^{2j+1}\,. \tag{21}$$

Now, we use the above mentioned formulae for finding a rational parameterization of n_1, n_2, h. We denote $\psi = \theta/b$. Using formulae (18), (19), taken for $p = 1$ and $q = b$, we obtain

$$n_1 = \sin\theta = \sum_{j=0}^{\lfloor \frac{b-1}{2} \rfloor} (-1)^j \binom{b}{2j+1} [\cos\psi]^{b-2j-1} [\sin\psi)]^{2j+1}\,, \tag{22}$$

$$n_2 = \cos\theta = \sum_{j=0}^{\lfloor \frac{b}{2} \rfloor} (-1)^j \binom{b}{2j} [\cos\psi]^{b-2j} [\sin\psi)]^{2j}\,. \tag{23}$$

Furthermore, if we apply (20), (21) on $h_3 = h_1 + h_2$, taken for $p = \bar{a}_1, \bar{a}_2$, we get

$$\begin{aligned} h_3 &= [A_1 \cos(\bar{a}_1\psi) + B_1 \sin(\bar{a}_1\psi)] + [A_2 \cos(\bar{a}_2\psi) + B_2 \sin(\bar{a}_2\psi)] = \\ &= \Bigg[A_1 \sum_{j=0}^{\lfloor \frac{\bar{a}_1}{2} \rfloor} (-1)^j \binom{\bar{a}_1}{2j} [\cos\psi]^{\bar{a}_1-2j} [\sin\psi)]^{2j} + \\ &\quad + B_1 \sum_{j=0}^{\lfloor \frac{\bar{a}_1-1}{2} \rfloor} (-1)^j \binom{\bar{a}_1}{2j+1} [\cos\psi]^{\bar{a}_1-2j-1} [\sin\psi)]^{2j+1} \Bigg] + \\ &\quad + \Bigg[A_2 \sum_{j=0}^{\lfloor \frac{\bar{a}_2}{2} \rfloor} (-1)^j \binom{\bar{a}_2}{2j} [\cos\psi]^{\bar{a}_2-2j} [\sin\psi)]^{2j} + \\ &\quad + B_2 \sum_{j=0}^{\lfloor \frac{\bar{a}_2-1}{2} \rfloor} (-1)^j \binom{\bar{a}_2}{2j+1} [\cos\psi]^{\bar{a}_2-2j-1} [\sin\psi)]^{2j+1} \Bigg]. \end{aligned} \tag{24}$$

Finally, after setting

$$\cos\psi = \frac{1-t^2}{1+t^2} \quad \text{and} \quad \sin\psi = \frac{2t}{1+t^2} \tag{25}$$

to (22), (23) and (24), we arrive at a rational parametrization of n_1, n_2 and h_3. This completes the proof. □

Clearly, the previous result can be immediately extended to all curves supported by (11).

Corollary 1. *All curves given by the support function (11) are rational Pythagorean hodograph curves.*

4 Interpolating G^2 Hermite Data with Convolutions of HE-Cycloids

In this section we will present an algorithm for G^2 Hermite interpolation with arcs of convolutions of HE-cycloids. G^2 Hermite data consists of two points, two unit tangent/normal vectors and two signed curvatures at those points. In general, Hermite interpolation is useful for approximations of general curves by spline curves with a given continuity. Input data for Hermite interpolation are sampled from a given curve and the segments between two consecutive sample points are approximated by arcs of a given type.

As shown in [27], translations, rotations and scalings of canonical HE-cycloids are described by four coefficients which appear linearly in the support function representation of general HE-cycloids. For G^2 Hermite interpolation, six input data are given, so it is necessary to introduce two new free parameters. This can be done by taking convolutions of two canonical HE-cycloids $\mathcal{C}_{b_1}^{a_1}$ and $\mathcal{C}_{b_2}^{a_2}$ (a_1, b_1, a_2, b_2 are arbitrary but fixed).

It was proved in Section 3.1 that the curve given by the support function

$$h(\theta) = v_x \sin\theta + v_y \cos\theta + s_1 \sin\frac{a_1}{b_1}\theta + c_1 \cos\frac{a_1}{b_1}\theta + s_2 \sin\frac{a_2}{b_2}\theta + c_2 \cos\frac{a_2}{b_2}\theta \quad (26)$$

is a translated convolution of scaled and rotated HE-cycloids $\mathcal{C}_{b_1}^{a_1}$ and $\mathcal{C}_{b_2}^{a_2}$ and all such transformations can be obtained by a suitable choice of coefficients v_x, v_y, s_1, c_1, s_2, $c_2 \in \mathbb{R}$. The curve with the support function (26) has the parameterization

$$\mathbf{x}(\theta) = h(\theta)(\sin\theta, \cos\theta)^\top + h'(\theta)(\cos\theta, -\sin\theta)^\top \quad (27)$$

and the signed curvature radius of this curve is given by

$$R(\theta) = -h(\theta) - h''(\theta). \quad (28)$$

Thus, coefficients $v_x, v_y, s_1, c_1, s_2, c_2$ appear in (27) and (28) linearly. This means that for given G^2 Hermite boundary data, it is possible to find coefficients $v_x, v_y, s_1, c_1, s_2, c_2$ by solving a system of linear equations, so that (a segment of) convolution of scaled and rotated $\mathcal{C}_{b_1}^{a_1}$ and $\mathcal{C}_{b_2}^{a_2}$ interpolates the given data.

Let us suppose that we are given G^2 Hermite input data, i.e., two distinct points $\mathbf{P}_0 = [x_0, y_0]^\top$ and $\mathbf{P}_1 = [x_1, y_1]^\top$ with associated unit normal vectors $\mathbf{n}_i = (\sin\theta_i, \cos\theta_i)^\top$, $i = 0, 1$ and signed curvatures κ_0 and κ_1. We distinguish the following two cases:

1. *Both κ_0 and κ_1 are non-zero and have the same sign, i.e., $\kappa_0\kappa_1 > 0$*

   This represents a general case. For such data we obtain four linear equations for given points and normal vectors using (27) and additional two linear equations for given curvature radii $1/\kappa_i, i = 0, 1$ using (28). The system for unknown coefficients

$v_x, v_y, s_1, c_1, s_2, c_2$ can be written in the following form

$$\begin{pmatrix} 1 & 0 & k_{b_1}^{a_1}(\theta_0) & l_{b_1}^{a_1}(\theta_0) & k_{b_2}^{a_2}(\theta_0) & l_{b_2}^{a_2}(\theta_0) \\ 0 & 1 & m_{b_1}^{a_1}(\theta_0) & n_{b_1}^{a_1}(\theta_0) & m_{b_2}^{a_2}(\theta_0) & n_{b_2}^{a_2}(\theta_0) \\ 1 & 0 & k_{b_1}^{a_1}(\theta_1) & l_{b_1}^{a_1}(\theta_1) & k_{b_2}^{a_2}(\theta_1) & l_{b_2}^{a_2}(\theta_1) \\ 0 & 1 & m_{b_1}^{a_1}(\theta_1) & n_{b_1}^{a_1}(\theta_1) & m_{b_2}^{a_2}(\theta_1) & n_{b_2}^{a_2}(\theta_1) \\ 0 & 0 & p_{b_1}^{a_1}(\theta_0) & q_{b_1}^{a_1}(\theta_0) & p_{b_2}^{a_2}(\theta_0) & q_{b_2}^{a_2}(\theta_0) \\ 0 & 0 & p_{b_1}^{a_1}(\theta_1) & q_{b_1}^{a_1}(\theta_1) & p_{b_2}^{a_2}(\theta_1) & q_{b_2}^{a_2}(\theta_1) \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ s_1 \\ c_1 \\ s_2 \\ c_2 \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ x_1 \\ y_1 \\ 1/\kappa_0 \\ 1/\kappa_1 \end{pmatrix}, \tag{29}$$

where

$$\begin{aligned} k_b^a(\theta) &= \cos(\theta)\cos\left(\tfrac{a}{b}\theta\right) + \tfrac{a}{b}\sin(\theta)\sin\left(\tfrac{a}{b}\theta\right), \\ l_b^a(\theta) &= \cos(\theta)\sin\left(\tfrac{a}{b}\theta\right) - \tfrac{a}{b}\cos\left(\tfrac{a}{b}\theta\right)\sin(\theta), \\ m_b^a(\theta) &= \cos\left(\tfrac{a}{b}\theta\right)\sin(\theta) - \tfrac{a}{b}\cos(\theta)\sin\left(\tfrac{a}{b}\theta\right), \\ n_b^a(\theta) &= \sin(\theta)\sin\left(\tfrac{a}{b}\theta\right) + \tfrac{a}{b}\cos(\theta)\cos\left(\tfrac{a}{b}\theta\right), \\ p_b^a(\theta) &= \tfrac{b^2-a^2}{b^2}\cos\left(\tfrac{a}{b}\theta\right), \\ q_b^a(\theta) &= \tfrac{b^2-a^2}{b^2}\sin\left(\tfrac{a}{b}\theta\right). \end{aligned}$$

It is obvious that the coefficient matrix of the system (29) is singular for $\theta_0 = \theta_1$, i.e., if the normal vectors associated to $\mathbf{P}_0$ and $\mathbf{P}_1$ have the same direction and orientation (in what follows we omit this degenerated case). Otherwise, we can always choose sample points sufficiently close to each other to guarantee that the matrix is regular and hence (29) has exactly one solution.

2. *Either* $\kappa_0 = 0$*, or* $\kappa_1 = 0$.

This means that one of the input points $\mathbf{P}_0$, or $\mathbf{P}_1$ is an inflection point of the approximated curve. Since convolutions of HE-cycloids have no inflection points, it is not possible to find an interpolating arc for such G^2 Hermite data. Without loss of generality, let us suppose that $\mathbf{P}_0$ corresponds to the inflection point and, thus, $\kappa_0 = 0$. In this case, it is enough to consider only an offset of a rotated and scaled HE-cycloid $\mathcal{C}_b^a$ given by the support function

$$h(\theta) = d + v_x \sin\theta + v_y \cos\theta + s \sin\frac{a}{b}\theta + c\cos\frac{a}{b}\theta. \tag{30}$$

Similarly to the previous case, we obtain four linear equations from (27) and one additional linear equation for a given curvature radius in the non-inflection point $\mathbf{P}_1$ from (30). The system for unknown coefficients v_x, v_y, s, c, d can be written in the following form

$$\begin{pmatrix} 1 & 0 & k_b^a(\theta_0) & l_b^a(\theta_0) & \cos(\theta_0) \\ 0 & 1 & m_b^a(\theta_0) & n_b^a(\theta_0) & \sin(\theta_0) \\ 1 & 0 & k_b^a(\theta_1) & l_b^a(\theta_1) & \cos(\theta_1) \\ 0 & 1 & m_b^a(\theta_1) & n_b^a(\theta_1) & \sin(\theta_1) \\ 0 & 0 & p_b^a(\theta_1) & q_b^a(\theta_1) & 1 \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ s \\ c \\ d \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ x_1 \\ y_1 \\ 1/\kappa_1 \end{pmatrix}. \tag{31}$$

Again, we assume that $\theta_0 \neq \theta_1$ and sample points to obtain a regular coefficient matrix of the system ([31]). Let us emphasize that the interpolant obtained by this method is connected to neighbouring arc of the spline curve at $\mathbf{P}_1$ with G^2 continuity and at $\mathbf{P}_0$ with only G^1 continuity.

To summarize, an approximation spline curve, which is obtained by our algorithm, is everywhere G^2 continuous except the inflection points, where it is only G^1 continuous. Moreover, after the reparameterization, see Section 3.2, we obtain a piecewise rational PH curve.

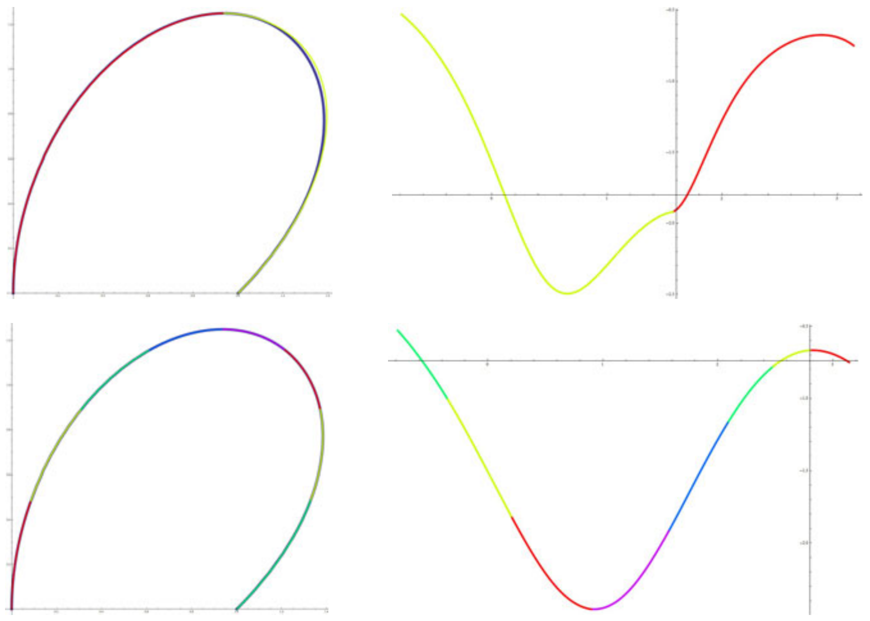


Fig. 3. Bézier curve and its approximation by a spline curve composed of arcs of the convolution of scaled and rotated $\mathcal{C}_3^1$ and $\mathcal{C}_2^1$ (left) and the curvature of the approximation spline curve (right)

Example 1. Let us consider two Bézier quartic curves on the interval $t \in [0, 1]$. The first curve has the control points $[0, 0]^\top$, $[0, 1]^\top$, $[1, 2]^\top$, $[2, 1]^\top$ and $[1, 0]^\top$ and no inflection points (see Fig. 3). The second curve has the control points $[0, 0]^\top$, $[2, 1]^\top$, $[2, -1]^\top$, $[3, 2]^\top$ and $[4, 0]^\top$ and two inflection points (see Fig. 4). In both cases, each curve segment without inflections is replaced by 2^N, $N = 1, \ldots, 10$, interpolating arcs of the convolution of $\mathcal{C}_3^1$ (the cardioid) and $\mathcal{C}_2^1$ (the nephroid). Table 1 summarizes the approximation error and its improvement (ratio of two consecutive errors). The error was obtained by sampling the Hausdorff distance. When no inflections are present, the approximation order is 6. At the inflections it drops to 3, which is due to the absence of inflection points on HE-cycloids and also on convolutions of HE-cycloid and G^1 continuity of the approximation at inflection points, cf. [27].

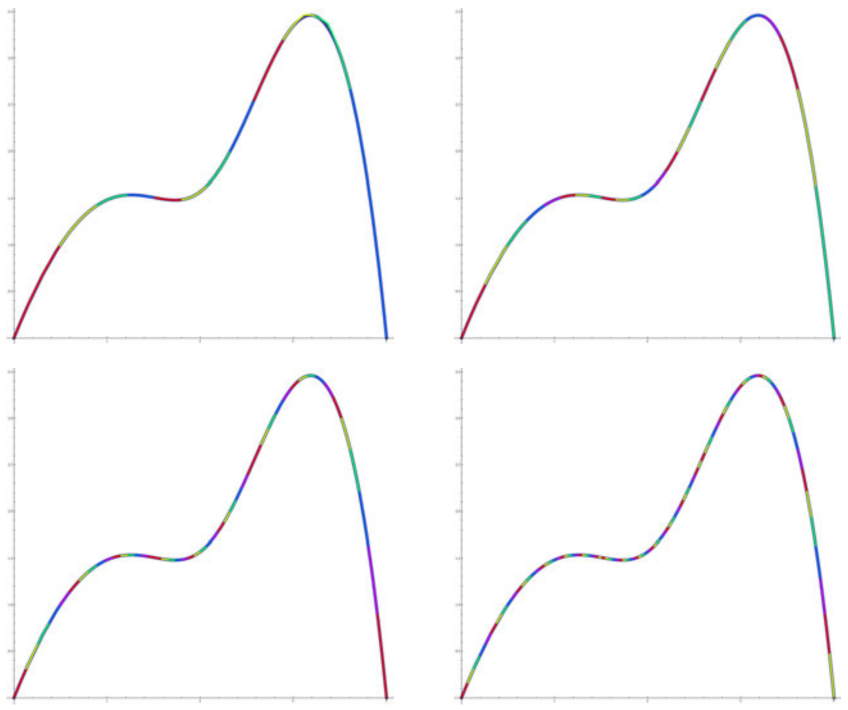


Fig. 4. Bézier curve with inflection points and its approximation by a spline curve composed of arcs of the convolution of $\mathcal{C}_3^1$ and $\mathcal{C}_2^1$

Table 1. Example for approximation by convolution of $\mathcal{C}_3^1$ and $\mathcal{C}_2^1$ conversion: Sampled errors and their improvements (ratios of two consecutive errors)

	No inflections		Inflections	
Parts	Error	Ratio	Error	Ratio
2	1.56137×10^{-2}		0.610492	
4	3.00896×10^{-4}	51.8907	2.56347×10^{-2}	23.815
8	5.10253×10^{-6}	58.9701	7.64343×10^{-4}	33.5383
16	8.22261×10^{-8}	62.0548	3.02679×10^{-5}	25.2526
32	1.30374×10^{-9}	63.0694	3.76836×10^{-6}	8.03212
64	2.04001×10^{-11}	63.9085	4.70233×10^{-7}	8.01381
128	3.19175×10^{-13}	63.9151	5.87338×10^{-8}	8.01381
256	4.98878×10^{-15}	63.9786	7.33906×10^{-9}	8.00290
512	7.79562×10^{-17}	63.9947	9.17222×10^{-10}	8.00140
1024	1.21809×10^{-18}	63.9987	1.14643×10^{-10}	8.00069

Hypo/epicycloids are special cases of the so called *hypo/epitrochoids* given by the expression

$$\mathbf{x}(\varphi) = (R - r) \cdot \mathbf{n}(\varphi) - d \cdot \mathbf{n}(k\varphi), \quad k = 1 - \frac{R}{r}, \tag{32}$$

where d is not necessarily equal to r, i.e., the tracing point $\mathbf{x}$ does not have to lie on the moving circle. Trochoidal profiles are widely used for designing volumetric machines (pumps, compressors, engines) and cycloidal speed reducers, cf. [3,26]. However, they are not generally PH and thus do not possess rational offsets, which is shown in the following example.

Example 2. Let us consider HE-trochoid given by (32) where $R = 3$, $r = 1$, $d = 2/5$. To prove that this HE-trochoid does not possess rational offsets, we need its rational parameterization

$$\mathbf{x}(t) = \left(\frac{4/5t(7+3t^2)}{(1+t^2)^2}, \frac{-4/5(-2+3t^4-3t^2)}{(1+t^2)^2} \right)^\top . \tag{33}$$

We find the corresponding implicit support function, cf. [20,1],

$$\begin{aligned} D_1(\mathbf{n}, h_1) = {} & -104544n_2^6 + 293868n_2^4n_1^2 - 718632n_2^2n_1^4 - 37044n_1^6 + 148500n_2^5h_1 - \\ & -297000n_2^3n_1^2h_1 - 445500n_2n_1^4h_1 - 37575n_2^4h_1^2 - 75150n_2^2n_1^2h_1^2 - \\ & -37575n_1^4h_1^2 - 25000n_2^3h_1^3 + 75000n_2n_1^2h_1^3 + 10000n_2^2h_1^4 + 10000n_1^2h_1^4 . \end{aligned}$$

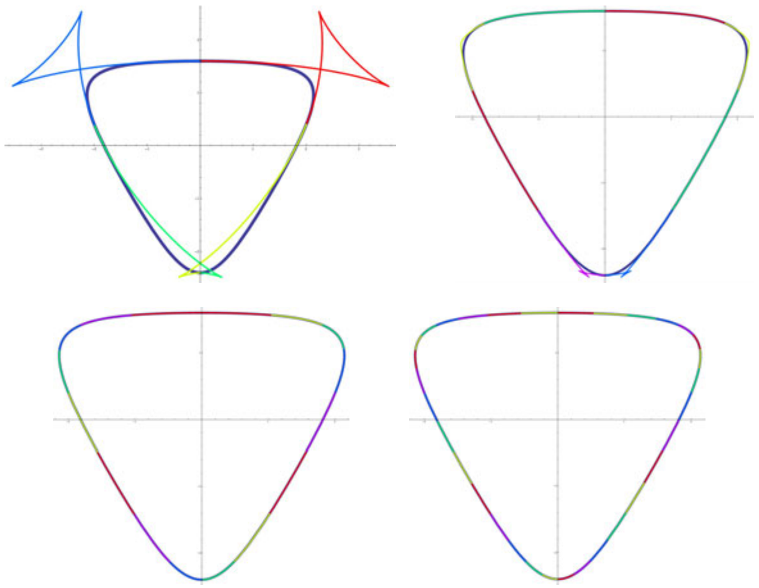


Fig. 5. Hypotrochoid ($R = 3, r = 1, d = 2/5$) and its approximation by spline curve composed of arcs of the convolution of $\mathcal{C}_3^1$ and $\mathcal{C}_2^1$

Next, we compute the implicit support function D_3 of the convolution of the hypotrochoid given by D_1 with the unit circle, see [20] for more details. Since the genus of D_3 is equal to 3, the curve is not rational which proves non-rationality of offsets of HE-trochoid given in this example (see Theorem 11 in [20]).

Finally, since this HE-trochoid is not a PH curve and it has no inflection points, we can construct its PH approximation with G^2 continuity everywhere using our method (see Fig. 5).

5 Conclusion

We have studied the Hermite interpolation of planar G^2 data with curves supported by trigonometric polynomials. It was shown that all curves of this type are nothing else but offsets to translated convolutions of scaled and rotated hypo/epicycloids. Their distinguished feature is the Pythagorean hodograph property, i.e., a rationality of their offsets.

The main advantage of the designed algorithm lies in the fact that the construction of the spline is based only on the solution of a simple system of (generally 6) linear equations. It was presented that an interpolant uniquely exists under some natural assumptions on the data (non-zero determinant of the system matrix) and the asymptotic approximation order 6 at non-inflection points was confirmed.

One can find some algorithms yielding C^2 (not just G^2) continuous *polynomial* PH spline curves (see e.g. [2,6,24]). However, as far as we are aware of the literature, this paper is the first paper which proposes a method for G^2 Hermite interpolation by *non-polynomial* PH curves.

Acknowledgement. The work on this paper was supported by the Research Plan MSM 4977751301. We thank all referees for their comments which helped us to improve the paper.

References

1. Aigner, M., Jüttler, B., Gonzales-Vega, L., Schicho, J.: Parameterizing surfaces with certain special support functions, including offsets of quadrics and rationally supported surface. Journal of Symbolic Computation 44(2), 180–191 (2009)
2. Albrecht, G., Farouki, R.T.: Construction of C^2 Pythagorean hodograph interpolating splines by the homotopy method. Adv. Comp. Math. 5, 417–442 (1996)
3. Bonandrini, G., Mimmi, G., Rottenbacher, C.: Theoretical analysis of internal epitrochoidal and hypotrochoidal machines. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 223(6), 1469–1480 (2009)
4. Choi, H.I., Han, C.Y., Moon, H.P., Roh, K.H., Wee, N.S.: Medial axis transform and offset curves by Minkowski Pythagorean hodograph curves. Computer-Aided Design 31(1), 59–72 (1999)
5. Farin, G.: Curves and surfaces for CAGD: a practical guide. Morgan Kaufmann Publishers Inc., San Francisco (2002)
6. Farouki, R.T., Kuspa, B.K., Manni, C., Sestini, A.: Efficient solution of the complex quadratic tridiagonal system for C^2 PH quintic splines. Numer. Alg. 27, 35–60 (2001)

7. Farouki, R.T., Rampersad, J.: Cycles upon cycles: An anecdotal history of higher curves in science and engineering. In: Dæhlen, M., Lyche, T., Schumaker, L.L. (eds.) Mathematical Methods for Curves and Surfaces II, pp. 95–116. Vanderbilt University Press (1998)
8. Farouki, R.: Pythagorean-Hodograph Curves: Algebra and Geometry Inseparable. Springer, Heidelberg (2008)
9. Goodman, T.N.T., Meek, D.S., Walton, D.J.: An involute spiral that matches G^2 Hermite data in the plane. Comput. Aided Geom. Des. 26(7), 733–756 (2009)
10. Gravesen, J., Jüttler, B., Šír, Z.: On rationally supported surfaces. Computer Aided Geometric Design 25, 320–331 (2008)
11. Habib, Z., Sakai, M.: Transition between concentric or tangent circles with a single segment of G^2 PH quintic curve. Comput. Aided Geom. Des. 25(4-5), 247–257 (2008)
12. Habib, Z., Sakai, M.: G^2 cubic transition between two circles with shape control. J. Comput. Appl. Math. 223(1), 133–144 (2009)
13. Jaklič, G., Kozak, J., Krajnc, M., Vitrih, V., Žagar, E.: On interpolation by planar cubic G^2 Pythagorean-hodograph spline curves. Mathematics of Computation (79), 305–326 (2010)
14. Jüttler, B.: Hermite interpolation by Pythagorean hodograph curves of degree seven. Math. Comp. 70, 1089–1111 (2001)
15. Kosinka, J., Jüttler, B.: G^1 Hermite interpolation by Minkowski Pythagorean hodograph cubics. Computer Aided Geometric Design 23, 401–418 (2006)
16. Kosinka, J., Jüttler, B.: C^1 Hermite interpolation by Pythagorean hodograph quintics in Minkowski space. Advances in Computational Mathematics 30, 123–140 (2009)
17. Kosinka, J., Lávička, M.: On rational Minkowski Pythagorean hodograph curves. Computer Aided Geometric Design 27(7), 514–524 (2010)
18. Kosinka, J., Šír, Z.: C^2 Hermite interpolation by Minkowski Pythagorean hodograph curves and medial axis transform approximation. Computer Aided Geometric Design 27(8), 631–643 (2010)
19. Lávička, M., Bastl, B.: Rational hypersurfaces with rational convolutions. Computer Aided Geometric Design 24(7), 410–426 (2007)
20. Lávička, M., Bastl, B., Šír, Z.: Reparameterization of curves and surfaces with respect to their convolution. In: Dæhlen, M., Floater, M., Lyche, T., Merrien, J.-L., Mørken, K., Schumaker, L.L. (eds.) MMCS 2008. LNCS, vol. 5862, pp. 285–298. Springer, Heidelberg (2010)
21. Meek, D.S., Walton, D.J.: Planar spirals that match G^2 Hermite data. Comput. Aided Geom. Des. 15(2), 103–126 (1998)
22. Meek, D.S., Walton, D.J.: Planar G^2 Hermite interpolation with some fair, C-shaped curves. J. Comput. Appl. Math. 139(1), 141–161 (2002)
23. Moon, H.: Minkowski Pythagorean hodographs. Computer Aided Geometric Design 16, 739–753 (1999)
24. Pelosi, F., Sampoli, M.L., Farouki, R.T., Manni, C.: A control polygon scheme for design of planar C^2 PH quintic spline curves. Computer Aided Geometric Design 24, 28–52 (2007)
25. Sabin, M.: A class of surfaces closed under five important geometric operations. Technical Report VTO/MS/207, British Aircraft Corporation (1974), http://www.damtp.cam.ac.uk/user/na/people/Malcolm/vtoms/vtos.html
26. Sánchez-Reyes, J.: Bézier representation of epitrochoids and hypotrochoids. Computer-Aided Design 31(12), 747–750 (1999)
27. Šír, Z., Bastl, B., Lávička, M.: Hermite interpolation by hypocycloids and epicycloids with rational offsets. Computer Aided Geometric Design 27, 405–417 (2010)
28. Šír, Z., Gravesen, J., Jüttler, B.: Computing convolutions and Minkowski sums via support functions. In: Chenin, P., et al. (eds.) Curve and Surface Design. Proceedings of the 6th International Conference on Curves and Surfaces, Avignon 2006, pp. 244–253. Nashboro Press (2007)

29. Šír, Z., Gravesen, J., Jüttler, B.: Curves and surfaces represented by polynomial support functions. Theoretical Computer Science 392(1-3), 141–157 (2008)
30. Vršek, J., Lávička, M.: On convolution of algebraic curves. Journal of symbolic computation 45(6), 657–676 (2010)
31. Walton, D.J., Meek, D.S.: G^2 curve design with a pair of Pythagorean Hodograph quintic spiral segments. Comput. Aided Geom. Des. 24(5), 267–285 (2007)
32. Walton, D.J., Meek, D.S.: G^2 Hermite interpolation with circular precision. Comput. Aided Des. 42(9), 749–758 (2010)
33. Yates, R.C.: Curves and their properties. National Council of Teachers of Mathematics (1974)

Approximating Algebraic Space Curves by Circular Arcs*

Szilvia Béla[1] and Bert Jüttler[2]

[1] Doctoral Program in Computational Mathematics
[2] Institute of Applied Geometry,
Johannes Kepler University, Altenberger Str. 69, 4040 Linz, Austria

Abstract. We introduce a new method to approximate algebraic space curves. The algorithm combines a subdivision technique with local approximation of piecewise regular algebraic curve segments. The local technique computes pairs of polynomials with modified Taylor expansions and generates approximating circular arcs. We analyze the connection between the generated approximating arcs and the osculating circles of the algebraic curve.

Keywords: algebraic space curve, circular arc, subdivision.

1 Introduction

Representing algebraic space curves is a fundamental problem of geometric computing. These curves are defined as the intersection curves of algebraic surfaces. Computation of such a surface-surface intersection is a basic operation in geometric modeling. It is important for evaluating set operations, for computing boundary curves and closely related to self-intersection problems. Several algorithms have been introduced to compute algebraic space curves. A survey of the topic is given by Patrikalakis and Maekawa [1].

Intersecting low degree implicitly defined surfaces has attracted a lot of interest in the literature. Quadratic surfaces are the simplest curved surfaces, therefore they are frequently used in computational geometry. The intersection computation of such surfaces has been discussed thoroughly in [2,3,4,5,6].

Several different methods have been developed for computing the intersection of algebraic surfaces (see [8,9,10]). Many of them are symbolic-numeric algorithms. The most widely used numeric methods are the lattice evaluation, tracing and subdivision-based methods. The lattice evaluation techniques solve a set of low dimensional sub-problems. Then the solution of these sub-problems is interpolated to approximate the general solution. Marching or tracing methods generate point sequences along the connected components of the curve. They necessarily use some topological information to find starting, turning and singular points [11,12,13,14]. Subdivision algorithms are based on the "divide and

* This work was supported by the Austrian Science Fund (FWF) through the Doctoral Program in Computational Mathematics, subproject 3.

conquer" paradigm. They decompose the problem into several sub-problems, and sort these problems according to the curve topology [15,16]. The decomposition terminates if suitable approximating primitives can be generated for each sub-problem [17]. In order to construct these approximating primitives, several local approximation techniques can be applied, such as interpolation, bounding region generation, least-squares approximation or Newton-type methods [18].

The existing local approximation techniques are based on the use of different types of approximating primitives. These primitives are often low degree curves since they have low computational costs. Several approximation algorithms generate line segments. However, algorithms that use quadratic curves may achieve a better convergence rate. A general quadratic curve represented in algebraic form can have self-intersection points or cups. In order to avoid to use non-regular curves as approximating primitives we generate circular arcs to approximate regular segments of the algebraic curve. As an alternative one can consider circular splines or bi-arcs as regular approximation primitives [23,24,25]. Circular arcs have many advantages compared to spline curves, such as exact arc length, offset and simple closest point computation. Therefore our local technique generates fat arcs (i.e., circular arcs with an error tolerance) as bounding primitives. The method we describe here can be used as a preprocessing step for approximating general curves by arc splines with a given tolerance, cf. [7].

In this paper we describe a hybrid algorithm which combines a subdivision technique with a new local approximation method. First we describe a technique for generating approximating circular arcs for regular algebraic curve segments. These arcs are closely related to the differential geometry of the algebraic curve. We discuss this relation and use it to confirm the convergence of the approximation technique. Then we present an algorithm using the arc generation technique combined with error estimation. In the end of the paper we describe how to combine the local arc generation approach with the global subdivision process and demonstrate the behavior of the algorithm by several examples.

2 Generating Approximating Circular Arcs

A segment of an algebraic space curve is given as the zero sets of two polynomials f and g in an axis-aligned box $\Omega_0 \subset \mathbb{R}^3$. It allocates the point set

$$\mathcal{C}(f, g, \Omega_0) = \{\mathbf{x} : f(\mathbf{x}) = 0 \wedge g(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega_0\}.$$

In order to use a local approximation method, we consider different segments of the curve in different sub-domains of the original box $\Omega \subset \Omega_0$. All these sub-domains are again assumed to be axis-aligned boxes. We would like to approximate segments of the curve by circular arcs. Clearly this is possible only for sub-domains, that do not contain singularities of the algebraic curve. In order to control this property on a certain domain we define regular points and regular segments of algebraic space curves as follows.

Definition 1. *A point* $\mathbf{p}$ *of an algebraic space curve* $\mathcal{K} = \mathcal{C}(f, g, \Omega) \subset \mathbb{R}^3$ *is called* **regular***, if the vectors* $\nabla f(\mathbf{p})$ *and* $\nabla g(\mathbf{p})$ *are linearly independent (and called* **singular** *otherwise). An* **algebraic space curve is regular** *in the sub-domain* Ω*, if any point of the segment in* Ω *is regular.*

We suppose that the sub-domain $\Omega \subseteq \Omega_0$, where we compute, contains only regular segments of the algebraic curve. Then it does not contain curve segments with self-intersections [16]. However, the domain can contain several segments (even closed loops) of the curve.

We present a local approximation technique. First it reformulates the algebraic equations, which define the curve. More precisely, we try to find a certain combination of the given polynomials f and g, that possesses a special Hessian matrix in the center point $\mathbf{c} = (c^x, c^y, c^z)$ of the sub-domain Ω. Such a new polynomial h can be defined as the combination

$$h = kf + lg, \tag{1}$$

where k and l are linear polynomials and $(x, y, z) \in \Omega$

$$k(x, y, z) = a + k_1(x - c^x) + k_2(y - c^y) + k_3(z - c^z),$$
$$l(x, y, z) = b + l_1(x - c^x) + l_2(y - c^y) + l_3(z - c^z).$$

The zero level set of the polynomial h

$$\mathcal{Z}(h, \Omega) = \{\mathbf{x} : h(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega\}$$

is a surface, which contains the algebraic curve defined by f and g

$$\mathcal{K} = \mathcal{C}(f, g, \Omega) \subseteq \mathcal{Z}(h, \Omega).$$

We choose the coefficients of k and l such that the Hessian of h is a scalar multiple of the identity matrix in the center of the domain $\mathbf{c}$

$$\mathrm{Hess}(h)(\mathbf{c}) = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{pmatrix}, \quad \lambda \in \mathbb{R}. \tag{2}$$

If such an h can be computed, then the quadratic Taylor expansion of h about $\mathbf{c}$ has a spherical zero level set. In order to find h, we solve a linear system with eight variables (the coefficients of k and l) and five equations, that can be deducted from (2)

$$\left.\begin{aligned} h_{xx}(\mathbf{c}) - h_{yy}(\mathbf{c}) &= 0 \\ h_{yy}(\mathbf{c}) - h_{zz}(\mathbf{c}) &= 0 \\ h_{xy}(\mathbf{c}) &= 0 \\ h_{yz}(\mathbf{c}) &= 0 \\ h_{xz}(\mathbf{c}) &= 0 \end{aligned}\right\} \tag{3}$$

where

$$h_{uv} = \frac{\partial h}{\partial u \partial v}, \quad u, v \in \{x, y, z\}.$$

If the system has full rank, then the solution set in the space of coefficients of k and l is three-dimensional. Therefore we choose two coefficients as parameters in advance. More precisely, we suppose that the value of the constant term of the polynomials k and l are arbitrary but fixed $(a, b) \in \mathbb{R}^2$ and different from zero ($a \neq 0$ and $b \neq 0$).

Lemma 1. *Given two polynomials f and g over the domain $\Omega \subset \mathbb{R}^3$. We suppose that*

$$\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\| \neq 0 \tag{4}$$

holds in the center $\mathbf{c}$ of the domain. Then for any pair of $(a, b) \in \mathbb{R}^2$, where $a \neq 0$ and $b \neq 0$, there exist an exactly one-dimensional family of non-trivial polynomials, k and l, such that $h = kf + lg$ satisfies (3).

Proof. The Hessian matrix of h can be expressed with the help of f, g, k and l as

$$\begin{aligned}\mathrm{Hess}(h)(\mathbf{c}) &= \nabla k(\mathbf{c})\nabla f(\mathbf{c})^{\mathrm{T}} + \nabla f(\mathbf{c})\nabla k(\mathbf{c})^{\mathrm{T}} + a\,\mathrm{Hess}(f)(\mathbf{c}) \\ &\quad + \nabla l(\mathbf{c})\nabla g(\mathbf{c})^{\mathrm{T}} + \nabla g(\mathbf{c})\nabla l(\mathbf{c})^{\mathrm{T}} + b\,\mathrm{Hess}(g)(\mathbf{c}).\end{aligned} \tag{5}$$

For any values of the parameters $a \neq 0$ and $b \neq 0$ the system (3) can be reformulated as

$$\mathbf{Ak} = \begin{pmatrix} f_x(\mathbf{c}) & -f_y(\mathbf{c}) & 0 & g_x(\mathbf{c}) & -g_y(\mathbf{c}) & 0 \\ 0 & f_y(\mathbf{c}) & -f_z(\mathbf{c}) & 0 & g_y(\mathbf{c}) & -g_z(\mathbf{c}) \\ f_y(\mathbf{c}) & f_x(\mathbf{c}) & 0 & g_y(\mathbf{c}) & g_x(\mathbf{c}) & 0 \\ 0 & f_z(\mathbf{c}) & f_y(\mathbf{c}) & 0 & g_z(\mathbf{c}) & g_y(\mathbf{c}) \\ f_z(\mathbf{c}) & 0 & f_x(\mathbf{c}) & g_z(\mathbf{c}) & 0 & g_x(\mathbf{c}) \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ l_1 \\ l_2 \\ l_3 \end{pmatrix} = \mathbf{b}, \tag{6}$$

where the vector of constants is

$$\mathbf{b} = -a \begin{pmatrix} \frac{1}{2}(f_{xx}(\mathbf{c}) - f_{yy}(\mathbf{c})) \\ \frac{1}{2}(f_{yy}(\mathbf{c}) - f_{zz}(\mathbf{c})) \\ f_{xy}(\mathbf{c}) \\ f_{yz}(\mathbf{c}) \\ f_{xz}(\mathbf{c}) \end{pmatrix} - b \begin{pmatrix} \frac{1}{2}(g_{xx}(\mathbf{c}) - g_{yy}(\mathbf{c})) \\ \frac{1}{2}(g_{yy}(\mathbf{c}) - g_{zz}(\mathbf{c})) \\ g_{xy}(\mathbf{c}) \\ g_{yz}(\mathbf{c}) \\ g_{xz}(\mathbf{c}) \end{pmatrix}.$$

In order to be certain that the system (6) has a one-dimensional solution set, we have to show that the matrix $\mathbf{A}$ has rank 5. Therefore we analyze the 5×5 sub-matrices of $\mathbf{A}$. We denote with $\mathbf{A}_i$ the matrix, which we derive from $\mathbf{A}$ by deleting ith column. The determinants of the matrices $\mathbf{A}_4$, $\mathbf{A}_5$ and $\mathbf{A}$ are

$$\begin{aligned}\det(\mathbf{A}_4) &= -f_x(\mathbf{c})\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\|^2, \\ \det(\mathbf{A}_5) &= f_y(\mathbf{c})\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\|^2, \\ \det(\mathbf{A}_6) &= -f_z(\mathbf{c})\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\|^2.\end{aligned}$$

We know that $\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\| \neq 0$. This also implies, that one of the coordinates of $\nabla f(\mathbf{c})$, $f_x(\mathbf{c}), f_y(\mathbf{c})$ or $f_z(\mathbf{c})$ is non-zero. Consequently, that one of the determinants of $\mathbf{A}_4, \mathbf{A}_5$ or $\mathbf{A}_6$ is not zero. So $\mathbf{A}$ always has a full rank 5. The solution of the system $\mathbf{Ak} = \mathbf{b}$ exists, and it is a one-dimensional subspace in $\mathbb{R}^6$. □

According to Lemma 1, for any pair of (a, b) where $a \neq 0$ and $b \neq 0$, there exists a one-parameter family of polynomials k and l, such that $kf + lg$ satisfies (3). From this family of polynomials we always choose the one, which minimizes the Euclidean norm

$$\|\mathbf{k}\|_2 \rightarrow \min \text{ subject to } \mathbf{Ak} = \mathbf{b}, \tag{7}$$

where $\mathbf{k} = (k_1, k_2, k_3, l_1, l_2, l_3)$ is the common coefficient vector of k and l. This guarantees that the solution behaves as numerically well as possible during the computations.

Given the polynomials f and g, a value of (a, b) and a center point $\mathbf{c}$ of the domain Ω, we define the function

$$\mathcal{F}(f, g, (a, b), \mathbf{c}) = h = kf + lg \tag{8}$$

according to the construction in Lemma 1 and the assumption (7).

Remark 1. Suppose that the right-hand side of the system (6), i.e. the vector $\mathbf{b}$, vanishes for a certain pair of (a, b). In this case the solution set of (6) is a line in $\mathbb{R}^6$, which passes through the origin. Then the linear combination $af + bg$ fulfills the condition (2). According to (7) we always choose the solution of the system (6), which has the smallest length. In this special case, both k and l are constants.

The polynomial $h = \mathcal{F}(f, g, (a, b), \mathbf{c})$ fulfills the special condition for the Hessian (2). Thus the quadratic Taylor expansion of h about $\mathbf{c}$ has a spherical zero level set

$$T^2_{\mathbf{c}} h(\mathbf{x}) = h(\mathbf{c}) + \nabla h(\mathbf{c})^T (\mathbf{x} - \mathbf{c}) + \frac{1}{2} h_{xx}(\mathbf{c})(\mathbf{x} - \mathbf{c})^T (\mathbf{x} - \mathbf{c}), \quad \forall \mathbf{x} \in \Omega. \tag{9}$$

If we compute two polynomials for two different pairs of parameters $(a, b) \neq (a', b')$

$$\hat{f} = \mathcal{F}(f, g, (a, b), \mathbf{c}) \quad \text{and} \quad \hat{g} = \mathcal{F}(f, g, (a', b'), \mathbf{c}), \quad \text{such that} \quad a, b, a', b' \neq 0,$$

then their quadratic Taylor expansions about $\mathbf{c}$ can be denoted by

$$p = T^2_{\mathbf{c}} \hat{f} \quad \text{and} \quad q = T^2_{\mathbf{c}} \hat{g}.$$

These two polynomials define the algebraic set

$$\mathcal{S} = \mathcal{C}(p, q, \Omega) = \{\mathbf{x} : p(\mathbf{x}) = 0 \wedge q(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega\}. \tag{10}$$

If this algebraic set is not empty and $p \neq q$, then it forms a circular arc. This arc can be used as an approximating circular arc of the curve $\mathcal{K} = \mathcal{C}(f, g, \Omega)$. Later on the error of the approximation is estimated by a distance bound of the algebraic curves $\hat{\mathcal{K}} = \mathcal{C}(\hat{f}, \hat{g}, \Omega)$ and $\mathcal{S}$.

3 Convergence of Approximating Arcs

We analyze in this section the behavior of the generated approximating arcs and its convergence properties.

3.1 Connection with the Osculating Circle

Now we suppose that the center of the computational domain Ω is a point of the algebraic curve $\mathcal{K} = \mathcal{C}(f, g, \Omega)$ defined by the polynomials f and g and that it is not an inflection point of the curve. If the center point is denoted by $\mathbf{c}$, then

$$f(\mathbf{c}) = g(\mathbf{c}) = 0. \tag{11}$$

This special case plays an important role during the computations, since later we would like to approximate the curve in such sub-domains of the original domain, which tightly enclose the algebraic curve.

For an arbitrary pair of parameters we compute a new polynomial as the combination of f and g as we defined in (8)

$$h = \mathcal{F}(f, g, (a, b), \mathbf{c}).$$

Consider the quadratic polynomial

$$s = T^2_{\mathbf{c}} h.$$

According to the assumption (11) the center of the domain is a point of the zero set of h and s

$$h(\mathbf{c}) = s(\mathbf{c}) = af(\mathbf{c}) + bg(\mathbf{c}) = 0. \tag{12}$$

Then the quadratic approximating polynomial s takes the following form

$$s(\mathbf{x}) = \nabla h(\mathbf{c})^{\mathrm{T}}(\mathbf{x} - \mathbf{c}) + \lambda(\mathbf{x} - \mathbf{c})^{\mathrm{T}}(\mathbf{x} - \mathbf{c}), \tag{13}$$

where

$$\nabla h(\mathbf{c}) = a\nabla f(\mathbf{c}) + b\nabla g(\mathbf{c}), \tag{14}$$

and

$$\mathrm{Hess}(h)(\mathbf{c}) = \lambda \mathbf{I}^3,$$

as in (2). Suppose that $\lambda \neq 0$, then $s = 0$ can be written in the form

$$\left\langle \mathbf{x} - \left(\mathbf{c} + \frac{1}{\lambda}\nabla h(\mathbf{c})\right), \mathbf{x} - \left(\mathbf{c} + \frac{1}{\lambda}\nabla h(\mathbf{c})\right)\right\rangle = \frac{|\nabla h(\mathbf{c})|^2}{\lambda^2}, \tag{15}$$

which is an equation of a sphere. Therefore the radius of this sphere can be computed as

$$r = \frac{|\nabla h(\mathbf{c})|}{\lambda}. \tag{16}$$

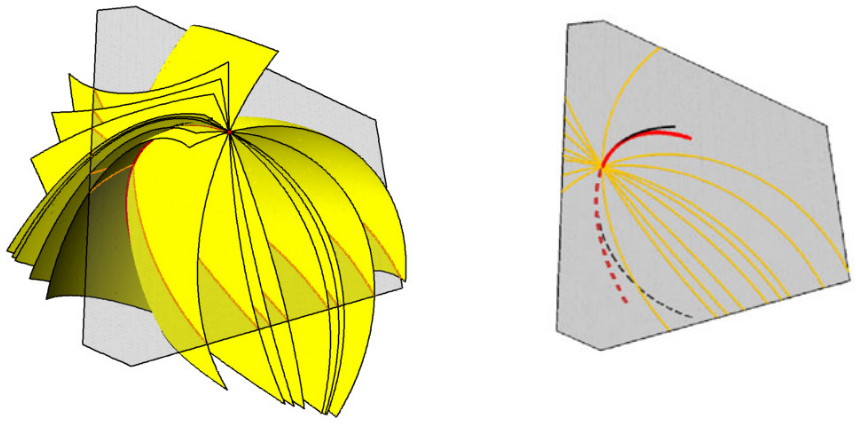


Fig. 1. Sphere family computed with Taylor expansion modification about a point on the algebraic curve (a) and its intersection with the normal plane of the curve (b). The thin, black curve is the algebraic curve. The red circle is the osculating circle.

Remark 2. The zero set of s defined in (13) depends only on the ratio of the chosen parameters a and b. Therefore the sphere family, computed for different values of (a, b), is a one-parametric surface family. It can be parametrized by the ratio of a and b. This follows from the computational method of k and l and the special form of the sphere equations (13). Fig. 1 (a) visualizes several members of such a sphere family for different values of a/b.

Lemma 2. *We assume that* (11) *is satisfied in the point* $\mathbf{c}$*. Then for each sphere equation, computed for a certain* $(a, b) \in \mathbb{R}^2$, $a, b \neq 0$*, the center of the sphere* $s = 0$ *lies in the normal plane of the algebraic curve in the point* $\mathbf{c}$*. Moreover the inverse of the radius of the sphere is exactly the normal curvature* κ_n *of the tangent direction* $\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})$ *of the surface* $\mathcal{F}(f, g, (a, b), \mathbf{c})$ *in the point* $\mathbf{c}$*.*

Proof. Suppose that in a certain neighborhood of the point $\mathbf{c}$ the algebraic curve can be parametrized with arc length parametrization. It is not a restriction, since we are computing only with the regular segments of the algebraic curve. The parametrization is denoted by

$$\mathbf{p}(t), \quad \text{where} \quad \mathbf{p}(t_0) = \mathbf{c}.$$

This curve lies on the surface $h = 0$ according to the definition, therefore it satisfies

$$\frac{d^i h(\mathbf{p}(t))}{dt^i} = 0,$$

for any i. If we compute the first derivative in the point $\mathbf{c}$, we obtain that

$$\left.\frac{dh(\mathbf{p}(t))}{dt}\right|_{t=t_0} = \langle \nabla h(\mathbf{c}), \mathbf{p}'(t_0)\rangle = 0.$$

Thus the tangent vector of the algebraic curve is parallel to the cross product of the gradients $\nabla f(\mathbf{c})$ and $\nabla g(\mathbf{c})$. In (14) we observed, that

$$\nabla h(\mathbf{c}) = a\nabla f(\mathbf{c}) + b\nabla g(\mathbf{c}).$$

Since s is the quadratic Taylor expansion of h about $\mathbf{c}$, we obtain that

$$\langle \nabla s(\mathbf{c}), \mathbf{p}'(t_0)\rangle = 0.$$

This implies, that for any values of the parameters (a, b) the gradient of the associated sphere is in the normal plane of the algebraic curve in the point $\mathbf{c}$.

The second derivative is

$$\left.\frac{d^2h(\mathbf{p}(t))}{dt^2}\right|_{t=t_0} = \langle \nabla h(\mathbf{c}), \mathbf{p}''(t_0)\rangle + \mathbf{p}'(t_0)^{\mathrm{T}}\mathrm{Hess}(h)(\mathbf{c})\mathbf{p}'(t_0) =$$

$$= \langle \nabla h(\mathbf{c}), \mathbf{p}''(t_0)\rangle + \lambda\langle \mathbf{p}'(t_0), \mathbf{p}'(t_0)\rangle = 0.$$

Since we used the arc length parametrization, therefore

$$\langle \nabla h(\mathbf{c}), \mathbf{p}''(t_0)\rangle + \lambda = 0.$$

The polynomial s is the quadratic Taylor expansion of h about $\mathbf{c}$, therefore also

$$\langle \nabla s(\mathbf{c}), \mathbf{p}''(t_0)\rangle = -\lambda.$$

If we expand the scalar product, then we get

$$|\nabla s(\mathbf{c})||\mathbf{p}''(t_0)| \cos\varphi = -\lambda,$$

where φ denotes the angle of the surface normal $\nabla h(\mathbf{c}) = \nabla s(\mathbf{c})$ and the normal direction of the algebraic curve in $\mathbf{c}$. According to the Theorem of Meusnier and (16) we finally arrive at

$$\kappa \cos\varphi = \kappa_n = \frac{\lambda}{|\nabla s(\mathbf{c})|} = \frac{1}{r},$$

which proves the lemma. $\square$

As an example, Fig. 1 (b) shows the intersection of the sphere family and the normal plane of the algebraic curve. Each sphere of the family intersects this plane in a great circle. These circles intersect each other in two points on the normal of the algebraic space curve. The second intersection point is not shown in the figure.

Corollary 1. *The functions f and g define an algebraic curve $\mathcal{K} = \mathcal{C}(f, g, \Omega)$ in $\Omega \subset \mathbb{R}^3$. We assume that the point $\mathbf{c} \in \Omega$ lies on the algebraic curve $\mathbf{c} \in \mathcal{K}$. We compute the function family $h(a, b) = \mathcal{F}(f, g, (a, b), \mathbf{c})$ with special Hessian for f and g in the point $\mathbf{c}$. The quadratic Taylor expansion for any (a, b) pair $a, b \neq 0$ has a spherical zero level set. The intersection of this sphere family is a circle, which is the osculating circle of $\mathcal{K}$ in the point $\mathbf{c}$.*

Proof. In each point of a curve on a surface, the osculating circle is the normal section of the curvature sphere of the surface [19]. In Lemma 2 we observed, that this curvature sphere for any $h(a, b) = 0$ surface is the zero set of the quadratic Taylor expansion. These spheres have the same intersection curve with the osculating plane of $\mathcal{K}$ in the point $\mathbf{c}$, which is exactly the osculating circle. □

3.2 Convergence of Approximating Circles

The method, which is described in Sect. 2, generates an approximation of the intersection curve of two algebraic surfaces f and g in the sub-domain $\Omega \subseteq \Omega_0$ by a circular arc. This approximating arc is defined as the intersection curve of two spheres. These spheres are given as the zero level set of two polynomials p and q. The polynomials are the quadratic Taylor expansion of certain polynomials with a special Hessian about the center point $\mathbf{c}$ of the domain Ω. The special polynomials are computed as the combination of the functions f and g in the form $kf + lg = \mathcal{F}(f, g, (a, b), \mathbf{c})$ for certain pair $a, b \neq 0$. Note that $\mathbf{c}$ is not required to lie on the curve segment.

In order to prove the convergence of the generated circles, we have to show that the computed polynomials depend continuously on the points of Ω_0 for a fixed choice of (a, b). It means, that the polynomial $\mathcal{F}(f, g, (a, b), \mathbf{c})$ depends continuously on the choice of the point $\mathbf{c}$.

Lemma 3. *Given two polynomials $f, g \in C^2$ over the domain Ω_0. We suppose that for any point $\mathbf{c} \in \Omega_0$*

$$\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\| \neq 0. \tag{17}$$

For an arbitrary but fixed pair of a and $b \in \mathbb{R} \setminus \{0\}$ we compute the polynomial

$$h = \mathcal{F}(f, g, (a, b), \mathbf{c})$$

with a special Hessian (see in Lemma 1) under the condition (7). Then h depends continuously on the points of the domain Ω_0.

Proof. We have to show that the computed linear factors k and l depend continuously on the point $\mathbf{c}$. We computed the coefficient vector $\mathbf{k} = (k_1, k_2, k_3, l_1, l_2, l_3)$, such that it satisfies the linear system $\mathbf{Ak} = \mathbf{b}$ in (6) and minimizes the l_2-norm of the vector $\mathbf{k}$ (see (7)). If (17) is true, then $\mathbf{A}$ has full rank in any point $\mathbf{c} \in \Omega_0$. For a full rank matrix the vector, which satisfies (6) and (7), can be computed as

$$\mathbf{k} = \underbrace{\mathbf{A}^{\mathrm{T}}(\mathbf{A}\mathbf{A}^{\mathrm{T}})^{-1}}_{\mathbf{A}^{\dagger}}\mathbf{b}.$$

The matrix $\mathbf{A}^\dagger$ is the so called Moore-Penrose generalized inverse of $\mathbf{A}$ (see [20]). If $f, g \in C^2$, then the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ depend continuously on the point $\mathbf{c}$. Therefore the vector $\mathbf{k}$ also depends continuously on the point $\mathbf{c}$. The values of $a \neq 0$ and $b \neq 0$ are fixed real numbers. So all coefficients a, b, k_i, $i = 1 \ldots 3$ and l_i, $i = 1 \ldots 3$ depend continuously on $\mathbf{c}$. Therefore also $kf + lg$ depends continuously on the point $\mathbf{c}$. □

The next corollary follows from Corollary 1 and Lemma 3. If we modify the Taylor expansion as described in Sect. 2, then we can establish a result concerning the behavior of a sequence of the generated approximating circles.

Corollary 2. *Suppose we have a nested sequence of sub-domains* $(\Omega_i)_{i=1,2,3\ldots} \subset \Omega_0$

$$\Omega_{i+1} \subset \Omega_i,$$

which have decreasing diameters δ_i*, such that*

$$\lim_{i\to\infty} \delta_i = 0,$$

and $\mathbf{c}_i$ *denotes the center point of* Ω_i*. Consider a pair of functions* f *and* g*, which defines an algebraic curve in* $\Omega_0 \subset \mathrm{I\!R}^3$

$$\mathcal{K}_0 = \mathcal{C}(f, g, \Omega_0) = \{\mathbf{x} : f(\mathbf{x}) = 0 \wedge g(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega_0\}.$$

Suppose that there exists a point $\mathbf{p}$*, which satisfies* $f(\mathbf{p}) = g(\mathbf{p}) = 0$ *and* $\mathbf{p} \in \Omega_i$ *for all* i*. We compute* $\hat{f}_i = \mathcal{F}(f, g, (a, b), \mathbf{c}_i)$ *and* $\hat{g}_i = \mathcal{F}(f, g, (a', b'), \mathbf{c}_i)$ *for fixed values of* $a, b, a', b' \neq 0$*. We consider the circles* $\mathcal{S}_i$ *defined by the zero set of the quadratic Taylor expansions* $p_i = T^2_{\mathbf{c}_i}\hat{f}_i$ *and* $q_i = T^2_{\mathbf{c}_i}\hat{g}_i$*. Then the sequence of these circles converges to a limit circle, which is the osculating circle of* $\mathcal{K}_0$ *in the point* $\mathbf{p}$*.*

The following corollary guarantees, that the local algebraic reformulation of the polynomials asymptotically does not influence the shape of the algebraic curve (no additional curve segment arises).

Corollary 3. *We assume that the parameters* a, b, a' *and* b' *are all non-zero and satisfy* $ab' \neq a'b$ *and that* $\mathbf{c}$ *varies in the compact domain* Ω_0*. There exists a constant* $d > 0$ *such that if the diameter* δ_Ω *of* $\Omega \subset \Omega_0$ *satisfies* $\delta_\Omega < d$ *and* $\mathbf{c} \in \Omega$ *then*

$$\mathcal{C}(f, g, \Omega) = \mathcal{C}(\hat{f}, \hat{g}, \Omega), \tag{18}$$

where $\hat{f} = \mathcal{F}(f, g, (a, b), \mathbf{c})$ *and* $\hat{g} = \mathcal{F}(f, g, (a', b'), \mathbf{c})$*.*

Proof. We write the algebraic reformulation as

$$\begin{pmatrix} \hat{f} \\ \hat{g} \end{pmatrix} = \begin{pmatrix} k_1 & l_1 \\ k_2 & l_2 \end{pmatrix} \begin{pmatrix} f \\ g \end{pmatrix} = \mathbf{L_c} \begin{pmatrix} f \\ g \end{pmatrix}.$$

Algorithm 1. `ArcLocal3d`$(f, g, \Omega, \varepsilon)$

```
Require: The curve is regular in $\Omega$.
 1: $\mathbf{c} \leftarrow$ center of $\Omega$
 2: $\hat{f} = \mathcal{F}(f, g, (a, b), \mathbf{c}), \quad \hat{g} = \mathcal{F}(f, g, (a', b'), \mathbf{c}), \quad a, b, a', b' \in \mathbb{R} \setminus \{0\}$
 3: $\mathcal{S} \leftarrow$ zero set of $T^2_{\mathbf{c}}\hat{f}$ and $T^2_{\mathbf{c}}\hat{g}$        {approximating circle}
 4: if $\mathcal{S} \neq \emptyset$ then
 5:    $G \leftarrow$ lower bound for $\|\nabla \hat{f}\|$ and $\|\nabla \hat{g}\|$
 6:    $K \leftarrow$ upper bound for $|\nabla \hat{f} \cdot \nabla \hat{g}|$
 7:    if $0 < G$ and $0 < G^2 - K$ then
 8:       $\varrho \leftarrow$ upper bound of $\mathrm{HD}_\Omega(\mathcal{S}, \mathcal{C}(\hat{f}, \hat{g}, \Omega))$        {see Sect. 4.2}
 9:       if $\varrho \leqslant \varepsilon$ then
10:          return $\mathcal{S} \cap \Omega$        {approximating arc has been found}
11:       end if
12:    end if
13: end if
14: return $\emptyset$        {no approximating arc has been found}
```

If $\det(\mathbf{L_c}(\mathbf{x})) \neq 0$ for all $\mathbf{x} \in \Omega$ then (18) is satisfied. (Note that this is merely a sufficient condition, but not a necessary one.) We know that

$$\mathbf{L_c}(\mathbf{c}) = ab' - a'b \neq 0.$$

According to Lemma 3 the linear polynomials $k_{1,2}$ and $l_{1,2}$ depend continuously on $\mathbf{c}$ and $\mathbf{x}$. Thus also $\nabla k_{1,2}$ and $\nabla l_{1,2}$ change continuously. Hence there exists a global upper bound B for all $\mathbf{x}$ and $\mathbf{c} \in \Omega_0$ such that

$$\|\nabla \det(\mathbf{L_c}(\mathbf{x}))\| < B.$$

Consequently, $\det(\mathbf{L_c}(\mathbf{x})) \neq 0$ is satisfied for all domains Ω containing $\mathbf{c}$ whose diameter does not exceed $|ab' - ba'|/(2B)$. $\square$

One might introduce an additionally test certifying that $\det(\mathbf{L_c})$ does not vanish in Ω. We omitted this test since we never experienced unwanted branches in our numerical experiments.

4 Algorithm and Error Bounds

Based on the results of Sect. 2 we formulate Algorithm 1. It generates an approximation of regular algebraic curve segments by circular arcs and a bound of the approximation error.

4.1 Local Algorithm

The local method is applied in such sub-domains of the original computational domain, which contains regular curve segments. Later on we will describe a global algorithm, which combines this local algorithm with subdivision method.

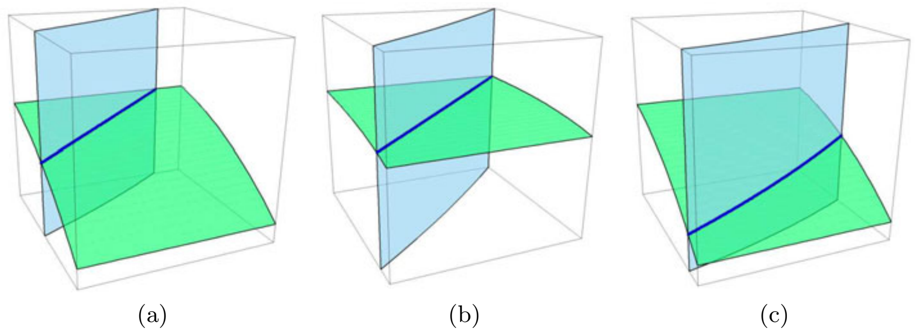


Fig. 2. Examples for local approximating arc generation with the help of `ArcLocal3d` for the parameter pairs $(a, b) = (1, 2)$ and $(a', b') = (2, 1)$

Table 1. Arc approximation of three different curve segments (see Fig. 2) for four different parameter choices

$(a, b); (a', b')$	Curve (a)	Curve (b)	Curve (c)
$(1, 2); (2, 1)$	0.01526	0.01184	0.01771
$(1, 5); (5, 1)$	0.01528	0.01177	0.01774
$(1, 100); (100, 1)$	0.01561	0.01151	0.01827
$(1, 1000); (1000, 1)$	0.01566	0.01150	0.01829

Remark 3. The algorithm `ArcLocal3d` first reformulates the polynomials. Therefore we have to fix the value of two parameter pairs (a, b) and (a', b'). According to Remark 2 if $a, b, a', b' \neq 0$ and $a/b \neq a'/b'$, then any choice of these parameter pairs generate similar results, since the generated approximating arcs converge to the same limit circle, the osculating circle. It is not possible to improve the general behavior of the algorithm by the choice of these parameters.

Fig. 2 presents three curve segments approximated by arcs with the help of the local algorithm for the parameter pairs $(a, b) = (1, 2)$ and $(a', b') = (2, 1)$. We approximate the same curve segments by three other choice of the parameter pairs in the unit cube. Table 1 compares the error bounds (see bounding method in Sect. 4.2) of these approximations. In all remaining examples, which will be presented in Section 5.2, we chose the parameters as $(1, 2)$ and $(2, 1)$.

The algorithm generates the approximating arc in algebraic form. Clearly, if we would like to represent the output in a parametric form, it is also possible to describe the circular arcs as rational quadratic curves.

The error estimation method (described in Sect. 4.2) is based on the convex hull property of polynomials represented in Bernstein-Bézier form. This technique is used to bound the distance between the zero level set of each polynomial and the associated quadratic Taylor expansion. Then an upper bound is generated for the one-sided Hausdorff distance of the approximating arc and the algebraic curve.

The algorithm is successful, if the approximating arc is found and the error bound is smaller then the prescribed tolerance ε. In this case the algorithm returns a circular arc, which approximates the curve segment in the appropriate sub-domain Ω. If the local algorithm fails then it returns the empty set.

4.2 Error Estimate

We describe a method here to estimate the distance of two algebraic space curves. In order to get a distance bound we combine a distance bound of parametric and algebraic curves and a distance estimation strategy between algebraic surfaces.

In order to bound the distance of algebraic and parametric curves we recall a result from [21]. We assume that the parametric curve segment $\mathbf{s}(t)$ is defined with the parameter domain $t \in [0,1]$ in $\Omega \subset \mathbb{R}^3$. The curve traces the point set

$$\mathcal{S} = \{\mathbf{s}(t) \,:\, t \in [0,1]\}.$$

The algebraic curve $\mathcal{K} = \mathcal{C}(f,g,\Omega)$ is defined as the simultaneous zero set of the polynomials f and g. The one-sided Hausdorff distance of $\mathcal{S}$ with respect to $\mathcal{K}^* = \mathcal{K} \cup \partial\Omega$ is defined as

$$\mathrm{HD}_\Omega(\mathcal{S},\mathcal{K}) = \sup_{t\in[0,1]} \inf_{\mathbf{x}\in\mathcal{K}^*} \|\mathbf{x} - \mathbf{s}(t)\|. \tag{19}$$

The boundary $\partial\Omega$ in (19) is needed for technical reasons, see [21].

Theorem 1 ([21]). *Suppose that the polynomials f and g define the algebraic curve $\mathcal{K} = \mathcal{C}(f,g,\Omega)$ in $\Omega \subset \mathbb{R}^3$. We assume that positive constants G and K exist for all $\mathbf{x} \in \Omega$, such that*

$$G \le \|\nabla f(\mathbf{x})\| \quad \textit{and} \quad G \le \|\nabla g(\mathbf{x})\|,$$

and

$$|\nabla f(\mathbf{x}) \cdot \nabla g(\mathbf{x})| \le K.$$

If

$$G^2 - K > 0,$$

and provided that there exist a positive constant M, such that

$$\forall t \in [0,1], \quad f(\mathbf{s}(t))^2 + g(\mathbf{s}(t))^2 \le M^2,$$

the one-sided Hausdorff distance is bounded by

$$\mathrm{HD}_\Omega(\mathcal{S},\mathcal{K}) \le \frac{M}{\sqrt{G^2 - K}}. \tag{20}$$

Theorem 1 can be used for bounding the distance of implicitly defined algebraic curves. The Bernstein-Bézier(BB) norm denoted as $\|.\|_{\text{BB}}^{\Omega}$, is the maximum absolute value of the coefficients in the BB-form of the polynomial represented in the domain Ω. With the help of this norm we can bound the distance of algebraic surfaces over a certain computational domain $\Omega \subset \mathbb{R}^3$. The distance bound can be defined between an arbitrary polynomial f and an approximating polynomial p for all point in the domain

$$\varepsilon = \|f - p\|_{\text{BB}}^{\Omega}. \tag{21}$$

Due to the convex hull property

$$|f(\mathbf{x}) - p(\mathbf{x})| \leq \varepsilon, \quad \forall \mathbf{x} \in \Omega.$$

Therefore for all $\mathbf{x} \in \Omega$ such that $p(\mathbf{x}) = 0$

$$\|f(\mathbf{x})\| \leq \varepsilon. \tag{22}$$

Suppose that an algebraic curve $\mathcal{K}$ is defined by the polynomials f and g

$$\mathcal{K} = \mathcal{C}(f, g, \Omega) = \{\mathbf{x} : f(\mathbf{x}) = 0 \wedge g(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega\}.$$

An approximating space curve $\mathcal{S}$ is given by two approximating algebraic surfaces $p = 0$ and $q = 0$

$$\mathcal{S} = \mathcal{C}(p, q, \Omega) = \{\mathbf{x} : p(\mathbf{x}) = 0 \wedge q(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega\}.$$

In order to estimate the distance of algebraic space curves we measure first the distance of the defining algebraic surfaces. Suppose that the polynomial p approximates f, and q is an approximating polynomial of g. We estimate the distance between the algebraic surfaces and the approximating surfaces pairwise

$$\varepsilon_1 = \|f - p\|_{\text{BB}}^{\Omega}, \quad \varepsilon_2 = \|g - q\|_{\text{BB}}^{\Omega}.$$

According to (22) for all $\mathbf{x} \in \mathcal{S}$

$$|f(\mathbf{x})| \leq \varepsilon_1 \quad \text{and} \quad |g(\mathbf{x})| \leq \varepsilon_2,$$

thus

$$\sqrt{f(\mathbf{x})^2 + g(\mathbf{x})^2} \leq \sqrt{\varepsilon_1^2 + \varepsilon_2^2}.$$

Therefore Theorem 1 can be applied to bound the distance of $\mathcal{K}$ and $\mathcal{S}$ with the help of the constants G, K and

$$M = \sqrt{\varepsilon_1^2 + \varepsilon_2^2}.$$

In order to compute the constants G, K, ε_1 and ε_2, we represent the polynomials in Bernstein-Bézier form and use the convex hull property of the representation form.

Remark 4. This error bound and the method for computing it can be extended to the two-sided Hausdorff distance. The role of the polynomials f, g and p, q can be exchanged in the bound (21). In this case, the values of the algebraic distances (ε_1 and ε_2) do not change. The bounds on the gradients G and K will change. However, both bounds converge to the same value when the domain Ω shrinks to a point. This follows from the construction of p and q, which satisfy

$$\nabla f(\mathbf{c}) = \nabla p(\mathbf{c}) \quad \text{and} \quad \nabla g(\mathbf{c}) = \nabla q(\mathbf{c})$$

in the center $\mathbf{c}$ of the computational domain.

5 Global Subdivision Method

Subdivision is a frequently used technique and it is often combined with local approximation methods. Such hybrid algorithms subdivide the computational domain in order to separate regions where the topology of the curve can be described easily. The local curve approximation techniques can be applied in the sub-domains, where the topology of the curve has been successfully analyzed. The regions with unknown curve behavior can be made smaller and smaller with subdivision.

5.1 Global Algorithm

The algorithm `GenerateArcs` (see Algorithm 2) generates approximating arcs for general algebraic space curves. It combines the arc generation for regular curve segments `ArcLocal3d` (see Algorithm 1) with recursive subdivision.

First it analyzes the Bernstein–Bézier coefficients of the polynomials with respect to the current sub-domain. If no sign changes are present for one or both of the polynomials, then the current sub-domain does not contain any component of the algebraic curve. Otherwise, if the curve is regular within the domain, it tries to apply the local arc generation algorithm. If this is not successful, then the algorithm either subdivides the current domain into eight sub-domains, or returns the entire domain, if its diameter is already below the user-defined threshold ε.

Note that the algorithm may return sub-domains that do not contain any segments of the implicitly defined curve. However, it is guaranteed to return a set of approximating primitives, such that each point of the implicitly defined curve has at most the distance ε to one of the primitives.

The generated approximating arcs are generally disconnected. It follows from the approximation technique described in the local algorithm `ArcLocal3d`. This technique always considers information only about the algebraic curve segment, which is located in the computational sub-domain. Due to the user-specified error bound ε, the end points of two neighboring approximating arcs along the curve cannot be farther away from each other than 2ε. However a post processing step can be applied in the end of the global algorithm to connect the approximating arcs, which represent the same segment of the curve. For instance based on the

Algorithm 2. `GenerateArcs`$(f, g, \Omega, \varepsilon)$

```
1: if the box is empty then
2:    return $\emptyset$
3: end if
4: if the curve is regular in $\Omega$ then
5:    $\mathcal{A} \leftarrow$ ArcLocal3d$(f, g, \Omega, \varepsilon)$                    {arc generation}
6:    if $\mathcal{A} \neq \emptyset$ then
7:       return $\mathcal{A}$                                            {... has been successful}
8:    end if
9: end if
10: if diameter of $\Omega > \varepsilon$ then
11:    subdivide the box into 8 subboxes $\Omega_1, \dots, \Omega_8$                    {subdivision}
12:    return $\bigcup_{i=1}^{8}$GenerateArcs$(f, \Omega_i, \varepsilon)$                 {recursive call}
13: end if
14: return $\Omega$                                           {current box is small enough}
```

error bound, one can apply approximation techniques, which generate continuous curves within tolerance bands [7].

5.2 Examples

Example 1. In this example we approximate the intersection curve of quadric surfaces. We apply the algorithm `GenerateArcs` for three different intersections of four different pairs of quadric surfaces. The outputs are represented in Fig. 3. The number of used approximating primitives are given in Tab. 2 for each intersection curve. If the curve has a singular point (here in 1.(b), 2.(c), 3.(b) and 4.(c)), then the algorithm returns not only arcs but also bounding boxes as approximating primitives. All the examples are represented in the unit cube $[0, 1]^3$. The intersection curves are approximated within the tolerance $\varepsilon = 0.01$.

According to the local approximation technique the global algorithm generates only bounding boxes around the singular points of the curve. If singular points are present, then our method generates bounding boxes, but no arcs. More sophisticated techniques are required for dealing with singular points, see e.g. [26]. Our further aim is to develop such computational tests based on arc approximation.

Example 2. In this example we approximate the intersection curve of non-quadratic surface patches by two different methods. One is using the circular arc generation technique combined with subdivision. The other technique generates only approximating line segments as local approximating primitives and combines it with iterative subdivision. This second method also reformulates the algebraic system locally. It generates two different linear combination of the algebraic equations such that the new polynomials have perpendicular normal vector in the center of the computational sub-domain. The approximating line segment is defined then as the intersection curve of the linear Taylor expansions of the reformulated polynomials.

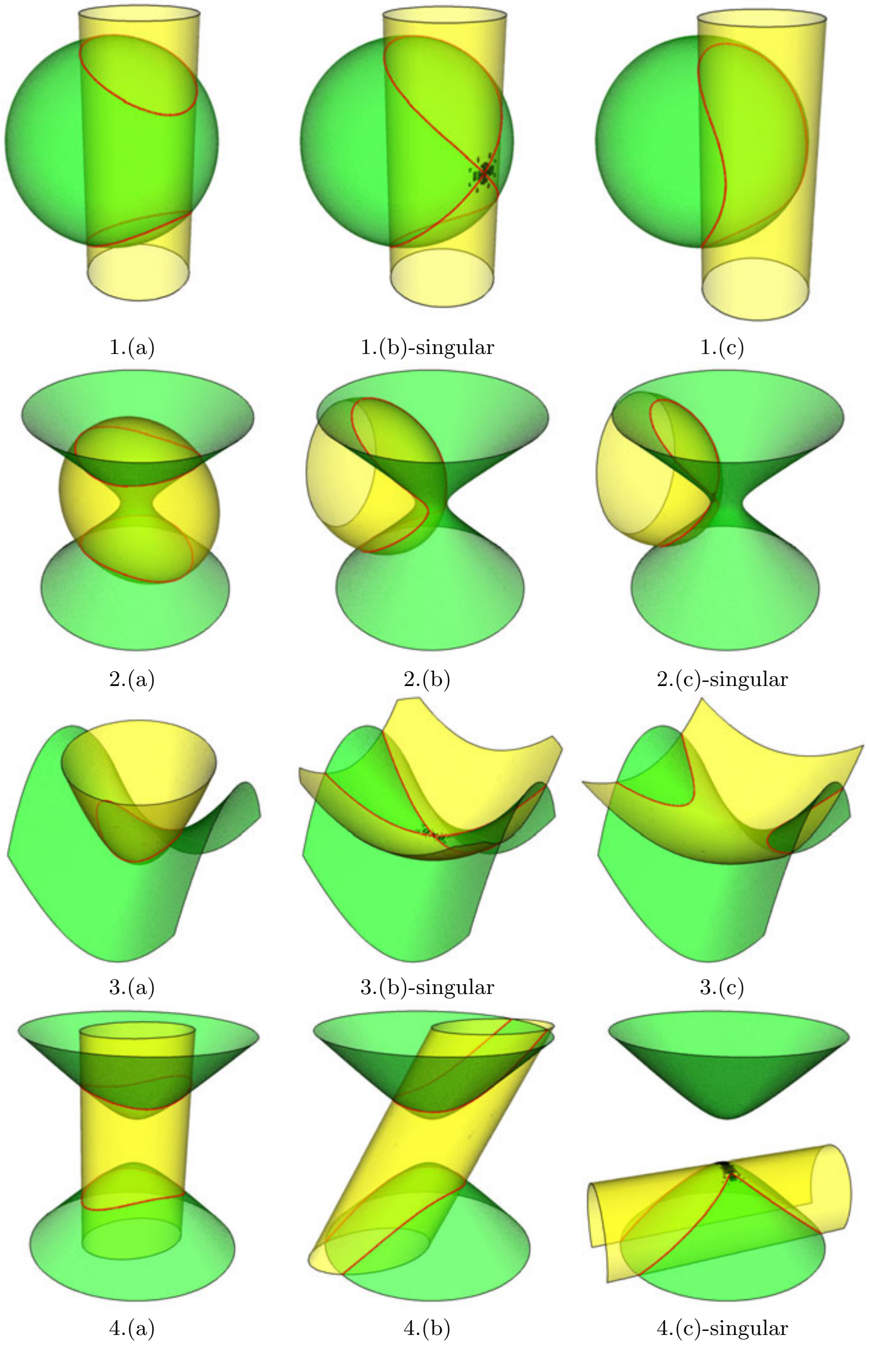


Fig. 3. Approximation of the intersection of quadric surfaces

Table 2. Approximating intersection curve of quadric surfaces. The number of used approximating primitives are given for the examples shown in Fig. 3.

Quadric Surfaces	Position (see Fig. 3)	Number of Arcs	Number of Boxes
1. sphere + cylinder	(a)	80	0
	(b)-singular	104	248
	(c)	52	0
2. ellipsoid + hyperboloid of one sheet	(a)	80	0
	(b)	76	0
	(c)-singular	96	76
3. rotational paraboloid + hyperbolic paraboloid	(a)	60	0
	(b)-singular	108	156
	(c)	50	0
4. hyperboloid of two sheets + elliptic cylinder	(a)	80	0
	(b)	80	0
	(c)-singular	88	612

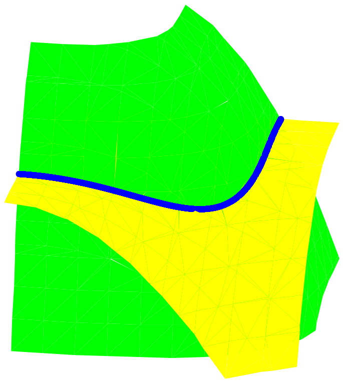


Fig. 4. Approximation of a high-order algebraic space curve by circular arcs (69 segments) and line segments (278 segments), both with tolerance $\varepsilon = 10^{-4}$. Only one picture is shown, since there are no visual differences.

The example surfaces are defined as the zero level set of the polynomials

$$\begin{aligned} 2x^4 + y^3 + z - 1.1 \\ x^3y^2 + z - 0.6 \end{aligned} \tag{23}$$

in the unit cube. The curve is approximated within the tolerance $\varepsilon = 10^{-4}$. The line approximation uses 278 line segments while the arc generation method only 69 arcs to reach the same tolerance level (see Fig. 4).

Example 3. In this example we approximate the isophotes of surfaces for different light directions. Isophotes are curves on a surface, where all points are exposed with equal light intensity from a given light source. An isophote of a surfaces $f = 0$ for a fixed direction vector $\mathbf{d}$ and angle φ traces the point set

$$\mathcal{I}(f, \mathbf{d}, \varphi) = \{\mathbf{p} : \; f(\mathbf{p}) = 0 \wedge \langle \mathbf{d}, \nabla f(\mathbf{p})\rangle = \cos(\varphi)\|\nabla f(\mathbf{p})\|\},$$

Table 3. Number of used approximating primitives (in columns # Arcs) in the isophote approximations (see examples in Fig.5 and Fig.6)

$\mathcal{S}_1 : xy - z + 0.5 = 0$						$\mathcal{S}_2 : x^3 + \frac{1}{2}y^3 + z - \frac{1}{2} = 0$			
$(0, 0, -1)$		$(-1, 1, -4)$		$(-2, 0, -3)$		$(-1, -1, -1)$		$(0, -1, -1)$	
$\cos\varphi$	# Arcs	$\cos\varphi$	# Arcs	$\cos\varphi$	# Arcs	$\cos\varphi$	# Arcs	$\cos\varphi$	# Arcs
0.8	**66**	0.7	**19**	0.5	**15**	0.6	**28**	0.3	**16**
0.85	**44**	0.8	**25**	0.65	**18**	0.7	**32**	0.4	**32**
0.9	**48**	0.88	**56**	0.8	**28**	0.75	**58**	0.5	**44**
0.95	**32**	0.95	**54**	0.9	**22**	0.8	**107**	0.7	**70**
0.99	**28**	0.99	**26**	0.97	**31**	0.85	**120**	0.99	**79**

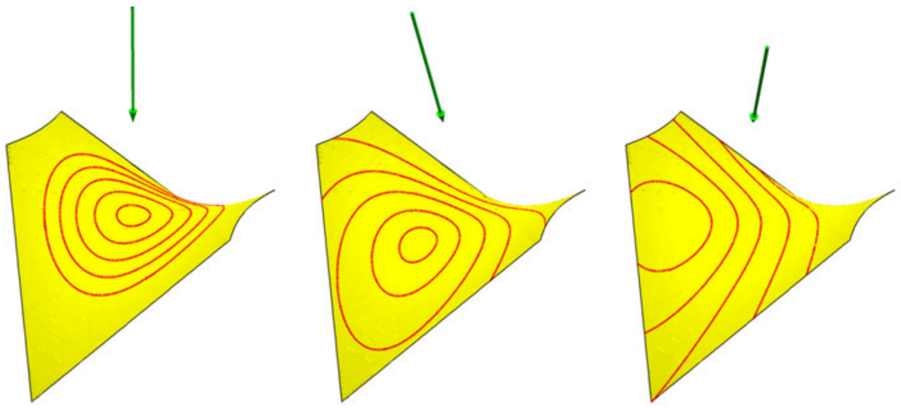


Fig. 5. Approximation of isophotes for different light directions on the surface $\mathcal{S}_1$

if we suppose that the direction vector is a unit vector. In order to describe an isophote for a given $\mathbf{d}$ and φ we used the algebraic equation system

$$\begin{aligned} f &= 0, \\ (f_x d^x + f_y d^y + f_z d^z)^2 - \cos^2\varphi\left(f_x^2 + f_y^2 + f_z^2\right) &= 0, \end{aligned}$$

where $\mathbf{d} = (d^x, d^y, d^z)$. These two equations allocates the points of the isophotes, which belong to the direction $\mathbf{d}$ and the angles φ and $(\pi - \varphi)$. We apply the arc approximation algorithm to two different surfaces to compute isophotes. For the first one we compute isophotes for three different light directions (see Fig. 5). For the second surface we used two different light directions (see Fig. 6). In Tab. 3 we show the number of used approximating arcs for each isophote for both surfaces, along with the light directions and angles. We approximated the isophotes in the domain $[-1, 1]^3$ within the tolerance $\varepsilon = 0.05$.

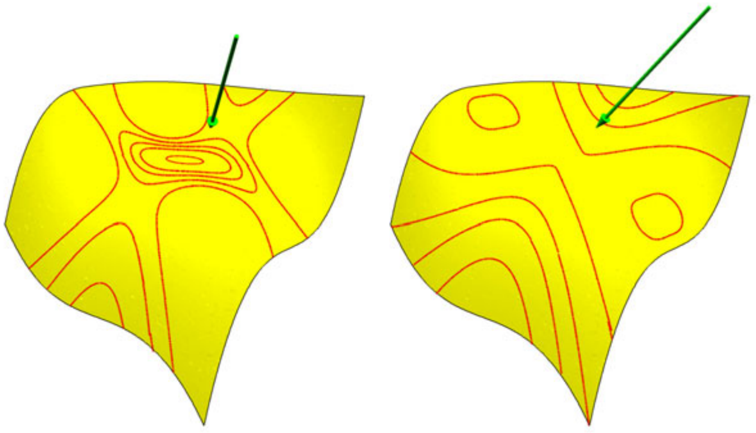


Fig. 6. Approximation of isophotes for different light directions on the surface $\mathcal{S}_2$

6 Conclusion

We presented a local technique which generates approximating circular arcs for regular segments of algebraic space curves. This technique is based on a reformulation of the problem. It combines the defining polynomials of the algebraic curve, such that the new polynomials define the same algebraic curve but they have special Hessian matrices at a certain point. This local technique can be combined with subdivision method to approximate arbitrary algebraic curves restricted to a domain.

A similar method can be applied also for planar algebraic curves. Moreover it might be possible to use a similar arc generation technique in $\mathbb{R}^n$ to approximate general algebraic curves. This result can be used also for finding roots of polynomial systems [17]. Computing with quadratic approximating primitives provides good convergence properties [22]. Thus it is promising to apply then for curve approximation and for solving polynomial systems.

References

1. Patrikalakis, N.M., Maekawa, T.: Shape interrogation for computer aided design and manufacturing. Springer, Heidelberg (2002)
2. Wang, W., Joe, B., Goldman, R.: Computing quadric surface intersections based on an analysis of plane cubic curves. Graphical Models 64, 335–367 (2002)
3. Wang, W., Goldman, R., Tu, C.H.: Enhancing Levin's method for computing quadric surface intersections. Computer Aided Geometric Design 20, 401–422 (2003)
4. Chau, S., Oberneder, M., Galligo, A., Jüttler, B.: Intersecting biquadratic surface patches. In: Piene, R., Jüttler, B. (eds.) Geometric Modeling and Algebraic Geometry, pp. 161–180. Springer, Heidelberg (2008)

5. Tu, C.H., Wang, W., Mourrain, B., Wang, J.Y.: Using signature sequences to classify intersection curves of two quadrics. Computer Aided Geometric Design 26, 317–335 (2009)
6. Dupont, L., Lazard, D., Lazard, S., Petitjean, S.: Near-optimal parameterization of the intersection of quadrics. In: SCG 2003: Proceedings of the Nineteenth Annual Symposium on Computational Geometry, pp. 246–255. ACM, New York (2003)
7. Held, M., Eibl, J.: Biarc approximation of polygons within asymmetric tolerance bands. Computer-Aided Design 37, 357–371 (2005)
8. Hoschek, J., Lasser, D.: Fundamentals of Computer Aided Geometric Design. AK Peters, Wellesley (1993)
9. Pratt, M.J., Geisow, A.D.: Surface/surface intersection problem. In: Gregory, J. (ed.) The Mathematics of Surfaces II, pp. 117–142. Claredon Press, Oxford (1986)
10. Patrikalakis, N.M.: Surface-to-Surface Intersections. IEEE Comput. Graph. Appl. 13(1), 89–95 (1993)
11. Bajaj, C.L., Hoffmann, C.M., Hopcroft, J.E., Lynch, R.E.: Tracing surface intersections. Comput. Aided Geom. Des. 5(4), 285–307 (1988)
12. Krishnan, S., Manocha, D.: An efficient surface intersection algorithm based on lower-dimensional formulation. ACM Trans. Graph. 16(1), 74–106 (1997)
13. Gonzalez-Vega, L., Necula, I.: Efficient topology determination of implicitly defined algebraic plane curves. Comput. Aided Geom. Design 19(9), 719–743 (2002)
14. Daouda, D.N., Mourrain, B., Ruatta, O.: On the computation of the topology of a non-reduced implicit space curve. In: Proc. Int. Symp. on Symbolic and Algebraic Computation, pp. 47–54. ACM, New York (2008)
15. Alcazar, J.G., Sendra, J.R.: Computation of the topology of real algebraic space curves. Journal of Symbolic Computation 39(6), 719–744 (2005)
16. Liang, C., Mourrain, B., Pavone, J.-P.: Subdivision methods for the topology of 2d and 3d implicit curves. In: Piene, R., Jüttler, B. (eds.) Geometric Modeling and Algebraic Geometry, pp. 199–214. Springer, Heidelberg (2008)
17. Mourrain, B., Pavone, J.P.: Subdivision methods for solving polynomial equations. Journal of Symbolic Computation 44(3), 292–306 (2009)
18. Elber, G., Kim, M.: Geometric constraint solver using multivariate rational spline functions. In: Proc. Symp. on Solid Modeling and Applications, pp. 1–10. ACM, New York (2001)
19. Kreyszig, E.: Differential Geometry. Dover, New York (1991)
20. Cline, R.E., Plemmons, R.J.: l_2-Solutions to Undetermined Linear Systems. SIAM Review 18(1), 92–106 (1976)
21. Jüttler, B., Chalmovianský, P.: A predictor–corrector technique for the approximate parameterization of intersection curves. Appl. Algebra Eng. Comm. Comp. 18, 151–168 (2007)
22. Sederberg, T.W., White, S.C., Zundel, A.K.: Fat arcs: a bounding region with cubic convergence. Comput. Aided Geom. Des. 6(3), 205–218 (1989)
23. Sabin, M.A.: The use of circular arcs to form curves interpolated through empirical data points. Technical Report VTO/MS/164, British Aircraft Corporation (1976)
24. Meek, D.S., Walton, D.J.: Approximating smooth planar curves by arc splines. Journal of Computational and Applied Mathematics 59, 221–231 (1995)
25. Song, X., Aigner, M., Chen, F., Jüttler, B.: Circular spline fitting using an evolution process. Journal of Computational and Applied Mathematics 231, 423–433 (2009)
26. Mantzaflaris, A., Mourrain, B.: Deflation and Certified Isolation of Singular Zeros of Polynomial Systems unpublished (2011), http://hal.inria.fr/inria-00556021/en

Generating Series for Drawing the Output of Dynamical Systems

Farida Benmakrouha, Christiane Hespel, and Edouard Monnier

INSA-IRISA,
20 avenue des Buttes de Coesmes, 35043 Rennes cedex, France
{farida.benmakrouha,christiane.hespel,edouard.monnier}@insa-rennes.fr

Abstract. We provide the drawing of the output of dynamical system (Σ), particularly when the output is rough or near instability points. (Σ) being analytical in a neighborhood of the initial state $q(0)$ and described by its state equations, its output $y(t)$ in a neighborhood of $t = 0$ is obtained by "evaluating" its generating series. Our algorithm consists in juxtaposing local approximating outputs on successive time intervals $[t_i, t_{i+1}]_{0 \leq i \leq n-1}$, to draw $y(t)$ everywhere as far as possible. At every point t_{i+1} we calculate at order k an approximated value of each component q_r of the state; on every interval $[t_i, t_{i+1}]_{0 \leq i \leq n-1}$ we calculate an approximated output. These computings are obtained from the symbolic expressions of the generating series of q_r and y, truncated at order k, specified for $t = t_i$ and "evaluated". A Maple package is built, providing a suitable result for oscillating outputs or near instability points when a Runge-Kutta method is wrong.

Keywords: Curve drawing, dynamical system, symbolic algorithm, generating series, oscillating output.

1 Introduction

The usual methods for drawing the output of dynamical systems consist in providing the drawing of a piecewise linear function or at best, a smooth sketch. These methods are based on an iterative construction of isolated points (Runge-Kutta).

The problem is that this drawing disregards or smooths the oscillations which may be interesting in physics or biology. For instance, the detection of climate cycles in climatology, the study of the synchronization of oscillations in biology and particularly the pharmacodynamy describing the effect of a drug on the organism, would be processed by the knowledge of an exact drawing of the model. Our idea is to obtain a drawing which preserves the oscillations as far as possible.

Rather than calculate numerous successive approximate points $y(t_i)_{i \in I}$, it can be interesting to provide some few successive local curves $\{y(t)\}_{t \in [t_i, t_{i+1}]_{0 \leq 1 \leq n-1}}$. In a previous paper [2], we provided a method for drawing the solution curve of differential equation. We had to compute the new values of the output and

its derivatives at every initial point $(t_i)_{0\leq 1\leq n-1}$. In the case when we consider a dynamical system, we have to calculate, for every interval $[t_i, t_{i+1}]$, the new state $q(t_i)$ instead of computing the new output $y(t_i)$ and its derivatives.
And then, we introduce the computing of the generating series $(G_{q_r,t})_{1\leq r\leq N}$ associated with the r^{th} component of the state $q = (q_1, \cdots, q_N)$ instead of only computing the generating series $G_{y,t}$ associated with the output.

The computing of these local curves can be kept partly generic since a generic expression of the generating series $G_{y,t}, (G_{q_r,t})_{1\leq r\leq N}$ of the system can be provided in terms of (q_r, t). The expression of the local curves $\{y(t)\}_{t\in[t_i,t_{i+1}]}$ is only a specification for $t = t_i$ at order k of the formula given in Section 3.

2 Preliminaries

2.1 Affine System, Generating Series

We consider the nonlinear analytical system affine in the input:

$$(\Sigma) \qquad \begin{cases} \dot{q} = f_0(q) + \sum_{j=1}^{m} f_j(q)u_j(t) \\ y(t) = g(q(t)) \end{cases} \tag{1}$$

– $(f_j)_{0\leq j\leq m}$ being some analytical vector fields in a neighborhood of $q(0)$
– g being the observation function analytical in a neighborhood of $q(0)$.

Its initial state is $q(0)$ at $t = 0$. The generating series $G_{y,0}$, in noncommutative variables, is built on the alphabet $Z = \{z_0, z_1, \cdots, z_m\}$, z_0 coding the drift and z_j coding the input $u_j(t)$. Generally $G_{y,0}$ is expressed as a formal sum $G_{y,0} = \sum_{w\in Z^*}\langle G_{y,0}|w\rangle w$ where $\langle G_{y,0}|z_{j_0}\cdots z_{j_l}\rangle = f_{j_0}\cdots f_{j_l}g(q)|_{q(0)}$ depends on $q(0)$.

2.2 Fliess's Formula and Iterated Integrals

The output $y(t)$ is given by the Fliess's equation ([4]):

$$y(t) = \sum_{w\in Z^*}\langle G_{y,0}|w\rangle \int_0^t \delta(w), \tag{2}$$

where $G_{y,0}$ is the generating series of (Σ) at $t = 0$:

$$\begin{aligned} G_{y,0} &= \textstyle\sum_{w\in Z^*}\langle G_{y,0}|w\rangle w \\ &= g(q)|_{q(0)} + \\ &\quad \textstyle\sum_{l\geq 0}\sum_{j_i=0}^{m} f_{j_0}\cdots f_{j_l}g(q)|_{q(0)} z_{j_0}\cdots z_{j_l}, \end{aligned} \tag{3}$$

and $\int_0^t \delta(w)$ is the iterated integral associated with the word $w \in Z^*$.
The iterated integral $\int_0^t \delta(w)$ of the word w for the input u is defined by

$$\begin{cases} \int_0^t \delta(\epsilon) &= 1 \\ \int_0^t \delta(vz_i) &= \int_0^t \left(\int_0^\tau \delta(v)\right) u_i(\tau)d\tau \\ & \qquad \forall z_i \in Z \quad \forall v \in Z^*, \end{cases} \tag{4}$$

where ϵ is the empty word, $u_0 \equiv 1$ is the drift and $u_{i\in[1..m]}$ is the ith input. We define the Chen's series as follows ([3]):

$$C_u(t) = \sum_{w\in Z^*} \int_0^t \delta(w). \tag{5}$$

We set

$$\xi_{i,1}(t) = \int_0^t u_i(\tau)d\tau. \tag{6}$$

From the previous definitions, we obtain the following expression:

$$y(t) = \sum_{w\in Z^*} \langle G_{y,0}|w\rangle\langle C_u(t)|w\rangle. \tag{7}$$

By applying the Fliess formula to every component $q_r(t)_{1\le r\le N}$ of the state $q(t)$ instead of $y(t)$, we obtain

$$q_r(t) = \sum_{w\in Z^*} \langle G_{q_r,0}|w\rangle\langle C_u(t)|w\rangle \tag{8}$$

by setting

$$\begin{aligned} G_{q_r,0} &= \textstyle\sum_{w\in Z^*} \langle G_{q_r,0}|w\rangle w \\ &= proj_r(q)|_{q(0)} + \\ &\quad \textstyle\sum_{l\ge 0}\sum_{j_i=0}^{m} f_{j_0}\cdots f_{j_l} proj_r(q)|_{q(0)} \\ &\quad z_{j_0}\cdots z_{j_l} \end{aligned} \tag{9}$$

where $proj_r(q) = q_r$.

3 Main Results

3.1 Approximate Values of the Output $y(t)$ and of the State $(q_r(t))_{1\le r\le N}$ in a Neighborhood of $t = 0$

Fliess's formula can be written as

$$y(t) = \langle G_{y,0}|\epsilon\rangle + \sum_{w\in Z^*-\{\epsilon\}} \langle G_{y,0}|w\rangle\langle C_u(t)|w\rangle. \tag{10}$$

An approximate function $y_k(t)$ of $y(t)$ up to order k in a neighborhood of $t = 0$ is obtained by expanding this expression up to the same order k. Then we have

$$|y(t) - y_k(t)| = O(t^{k+1}). \tag{11}$$

For instance, at order $k = 1$, $y(t)$ has the following approximate expression for a single input with drift

$$y_1(t) = \langle G_{y,0}|\epsilon\rangle + \langle G_{y,0}|z_0\rangle t + \langle G_{y,0}|z_1\rangle \xi_1(t), \tag{12}$$

where $\xi_k(t)$ denotes the kth primitive of $u(t)$.

This computation can be generalized $\forall r \in [1 \cdots N]$ components of the state q.

$$q_r(t) = \langle G_{q_r,0} | \epsilon \rangle + \sum_{w \in Z^* - \{\epsilon\}} \langle G_{q_r,0} | w \rangle \langle C_u(t) | w \rangle \tag{13}$$

by expanding these expressions up to order k.

3.2 Generalization at Time $t = t_i$

For a single input with drift, the system (Σ) can be written at $t = t_i$:

$$\begin{cases} \dot{q}(t_i + h) = f_0(q(t_i + h)) + \\ \qquad\qquad f_1(q(t_i + h))u(t_i + h) \\ y(t_i + h) = g(q(t_i + h)). \end{cases} \tag{14}$$

By setting

$$\begin{cases} U_i(h) &= u(t_i + h) \\ Y_i(h) &= y(t_i + h) \\ Q_i(h) &= q(t_i + h), \end{cases} \tag{15}$$

we obtain the following system:

$$(\Sigma_i) \qquad \begin{cases} \dot{Q}_i(h) = f_0(Q_i(h)) + f_1(Q_i(h)U_i(h) \\ Y_i(h) = g(Q_i(h), \end{cases} \tag{16}$$

and G_{y,t_i}, G_{q_r,t_i} are the generating series of (Σ_i). By setting $\psi_{i,k}(h) = \xi_k(t_i + h)$, then $\psi_{i,k}(h)$ is the kth primitive of $u(t_i + h)$ or the kth primitive of $U_i(h)$.

We have the equalities

$$\xi_1(t_i + h) = \int_{t_i}^{t_i+h} u(\tau)d\tau = \int_0^h U_i(t)dt = \psi_{i,1}(h), \tag{17}$$

and then, we can prove recursively that the Chen's integral $\int_{t_i}^{t_i+h} \delta(w)$ can be computed as an integral $\int_0^t \delta(W)$ by considering $U_i(t)$ instead of $u(t_i + t)$.

4 Application: For Drawing the Output

We present an application to the drawing of the output of a dynamical system. We consider the following dynamical system

$$\begin{cases} \dot{q}(t) = f_0(q(t)) + f_1(q(t))u(t) \\ y(t) = g(q(t)), \end{cases} \tag{18}$$

with initial conditions

$$q(0) = (q_1(0), \cdots, q_N(0)).$$

We propose a curve drawing of the output $y(t)$ of this system in $[0,T] = \bigcup[t_i, t_{i+1}]_{0\leq i\leq n-1}$ according to the following algorithm: Firstly, we compute a generic approximated expression of the generating series $G_{y,t}$, $(G_{q_r,t})_{1\leq r\leq N}$:

$$\begin{cases} G_{y,t} = \sum_{l\leq k} f_{j_0}\cdots f_{j_l} g(q)|_{q(t)} z_{j_0}\cdots z_{j_l} & = \sum_{|w|\leq k} \langle G_{y,t}|w\rangle w \\ G_{q_r,t} = \sum_{l\leq k} f_{j_0}\cdots f_{j_l} proj_r(q)|_{q(t)} z_{j_0}\cdots z_{j_l} & = \sum_{|w|\leq k} \langle G_{q_r,t}|w\rangle w. \end{cases} \tag{19}$$

We compute a generic approximated expressions of the output $y(t+h)$ and $q_r(t+h)$:

$$\begin{cases} y(t+h) = \sum_{|w|\leq k} \langle G_{y,t}|w\rangle \int_0^t \delta(W) \\ q_r(t+h) = \sum_{|w|\leq k} \langle G_{q_r,t}|w\rangle \int_0^t \delta(W) \end{cases} \tag{20}$$

– Initial point $t_0 = 0$:
  $q_1(0), \cdots, q_N(0)$ are given.
  The vector fields f_0, f_1 evaluated in t_0 provide $\langle G_{y,0}|w\rangle, (\langle G_{q_r,0}|w\rangle)_{1\leq r\leq N}$ for $|w| \leq k$ and $y(0) = g(q(0))$.

– Step i:
  Knowing $(q_r(t_{i-1}))_{1\leq r\leq N}, y(t_{i-1})$ and $(\langle G_{q_r,t_{i-1}}|w\rangle)_{1\leq r\leq N}, \langle G_{y,t_{i-1}}|w\rangle$ for $|w| \leq k$, we compute

$$\begin{cases} y(t_i) = y(t_{i-1}+h) \\ \qquad = \sum_{|w|\leq k} \langle G_{y,t_{i-1}}|w\rangle \int_0^h \delta(W) \\ q_r(t_i) = q_r(t_{i-1}+h) \\ \qquad = \sum_{|w|\leq k} \langle G_{q_r,t_{i-1}}|w\rangle \int_0^h \delta(W). \end{cases} \tag{21}$$

  We present $G_{y,t_i}, (G_{q_r,t_i})_{1\leq r\leq N}$ by evaluating their generic expressions at $q(t_i)$. We draw the local curve of the function $t_{i-1}+dt \longrightarrow y(t_{i-1}+dt)$ on the interval $[t_{i-1}, t_i]$.
– Final point $t = T = t_n$: stop at $i = n$.

4.1 Genericity of the Method

The computing of the coefficients

$$\begin{cases} \langle G_{y,t_i}|z_{j_0}\cdots z_{j_l}\rangle = f_{j_0}\cdots f_{j_l} g(q)|_{q(t_i)} \\ \langle G_{q_r,t_i}|z_{j_0}\cdots z_{j_l}\rangle = f_{j_0}\cdots f_{j_l} proj_r(q)|_{q(t_i)} \end{cases} \tag{22}$$

is generic. The computing of the expressions

$$\begin{cases} Y_i(h) & = y(t_i+h) = y(t_i) + \\ & \quad \sum_{|w|\leq k} \langle G_{y,t_i}|w\rangle\langle C_{U_i}(h)|w\rangle \\ q_r(t_i+h) & = q_r(t_i) + \\ & \quad \sum_{|w|\leq k} \langle G_{q_r,t_i}|w\rangle\langle C_{U_i}(h)|w\rangle \end{cases} \tag{23}$$

is also generic.

We use the previous algorithm by specifying t_i at every step in the previous expressions.

4.2 Example 1: Electric Differential Equation

Consider

$$\begin{array}{l} y^{(1)}(t) + k_1 y(t) + k_2 y^2(t) = u(t) \\ y(0) = y_0, \end{array} \tag{24}$$

which can be written as a first order differential system

$$\begin{cases} q^{(1)}(t) = -k_1 q(t) - k_2 q^2(t) + u(t) \\ \qquad\quad\; = a(q)(t) + u(t) \\ y(t) \quad = q(t), q(0) = y_0. \end{cases} \tag{25}$$

The vector fields are

$$\begin{array}{l} f_0(q) = -(k_1 q + k_2 q^2)\frac{d}{dq} = a(q)\frac{d}{dq} \\ f_1(q) = \frac{d}{dq}. \end{array}$$

1. Generic expression of $G_{y,t_i} = G_{q,t_i}$:
   Let us remark that
   $$\langle G_{y,t_i} | w z_1 \rangle = 0 \ \ \forall w \in Z^+.$$
   For instance, for order $k = 2$
   $$\begin{array}{ll} \langle G_{y,t_i} | \epsilon \rangle & = q|_{q(t_i)} \\ \langle G_{y,t_i} | z_0 \rangle & = a(q)|_{q(t_i)} \\ \langle G_{y,t_i} | z_1 \rangle & = 1 \\ \langle G_{y,t_i} | z_0^2 \rangle & = a(q)\frac{d}{dq}a(q)|_{q(t_i)} \\ \langle G_{y,t_i} | z_1 z_0 \rangle & = \frac{d}{dq}a(q)|_{q(t_i)}. \end{array} \tag{26}$$
2. Generic expression of $Y_i(h)$:
   for order $k = 2$
   $$\begin{aligned} Y_i(h) &= y(t_i + h) \\ &= y(t_i) + \\ &\quad \langle G_{y,t_i} | z_0 \rangle h + \langle G_{y,t_i} | z_1 \rangle \psi_{i,1}(h) + \\ &\quad \langle G_{y,t_i} | z_0^2 \rangle h^2/2 + \langle G_{y,t_i} | z_1 z_0 \rangle \psi_{i,2}(h). \end{aligned} \tag{27}$$
3. We use the previous algorithm of Section 4 by specifying t_i at every step. So we obtain the drawing of $y(t)$ (see the next section).

4.3 Example 2: Dynamical System with Polynomial Generating Series [7]

Consider

$$\begin{array}{ll} \dot{q}_1 & = q_2 + q_3 u(t) \\ \dot{q}_2 & = 1 \\ \dot{q}_3 & = u(t) \\ y(t) & = q_1(t) \\ y(0) & = q_1(0) = q_{1,0}, q_2(0) = q_{2,0}, q_3(0) = q_{3,0} \end{array} \tag{28}$$

which can be written as a dynamical system

$$(\Sigma) \qquad \begin{cases} \dot{q} = f_0(q) + f_1(q)u(t) \\ y(t) = g(q(t)) \end{cases} \tag{29}$$

for

$$\begin{aligned} f_0 &= q_2 \frac{\partial}{\partial q_1} + \frac{\partial}{\partial q_2} \\ f_1 &= q_3 \frac{\partial}{\partial q_1} + \frac{\partial}{\partial q_3} \\ y(t) &= proj_1(q(t)). \end{aligned}$$

1. Generic expression of $G_{y,t_i} = G_{q_1,t_i}$:
   This series is polynomial:

$$G_{q_1,t_i} = q_1(t_i) + q_2(t_i)z_0 + q_3(t_i)z_1 + z_0^2 + z_1^2. \tag{30}$$

2. Generic expression of G_{q_2,t_i}:
   This series is polynomial:

$$G_{q_2,t_i} = q_2(t_i) + z_0. \tag{31}$$

3. Generic expression of G_{q_3,t_i}:
   This series is polynomial:

$$G_{q_3,t_i} = q_3(t_i) + z_1. \tag{32}$$

4. Generic expression of $Y_i(h) = q_1(t_i + h)$:

$$\begin{aligned} Y_i(h) &= y(t_i + h) \\ &= y(t_i) + \\ &\quad \langle G_{y,t_i} | z_0 \rangle h + \langle G_{y,t_i} | z_1 \rangle \psi_{i,1}(h) + \\ &\quad \langle G_{y,t_i} | z_0^2 \rangle h^2/2 + \\ &\quad \langle G_{y,t_i} | z_1^2 \rangle 1/2(\psi_{i,1}(h))^2. \end{aligned} \tag{33}$$

5. Generic expression of $q_2(t_i + h)$:

$$q_2(t_i + h) = q_2(t_i) + h \tag{34}$$

6. Generic expression of $q_3(t_i + h)$:

$$q_3(t_i + h) = q_3(t_i) + \psi_{i,1}(h). \tag{35}$$

The drawing of $y(t)$ is exact for order ≥ 2 since the 3 generating series are polynomials of degree ≤ 2.

4.4 Example 3: Dynamical System [1]

Consider

$$\begin{aligned} \dot{q}_1 &= sin(q_2) \\ \dot{q}_2 &= sin(q_3) \\ \dot{q}_3 &= q_4^3 + u(t) \\ \dot{q}_4 &= -q_4^3 - q_1^{10} \\ y(t) &= q_2(t) \\ y(0) &= q_2(0) = q_{2,0}, q_1(0) = q_{1,0}, \\ &\quad q_3(0) = q_{3,0}, q_4(0) = q_{4,0} \end{aligned} \tag{36}$$

which can be written as a dynamical system

$$(\Sigma) \qquad \begin{cases} \dot{q} = f_0(q) + f_1(q)u(t) \\ y(t) = g(q(t)) \end{cases} \tag{37}$$

for

$$\begin{aligned} f_0 &= sin(q_2)\frac{\partial}{\partial q_1} + sin(q_3)\frac{\partial}{\partial q_2} + \\ &\quad q_4^3\frac{\partial}{\partial q_3} + (-q_4^3 - q_1^{10})\frac{\partial}{\partial q_4} \\ f_1 &= \frac{\partial}{\partial q_3} \\ y(t) &= proj_2(q(t)). \end{aligned}$$

1. Generic expression of $G_{y,t_i} = G_{q_2,t_i}$ up to order $k = 2$:

$$\begin{aligned} G_{q_2,t_i} = q_2(t_i) &+ sin(q_3(t_i))z_0 + \\ &(q_4(t_i))^3 cos(q_3(t_i))z_0^2 + \\ &cos(q_3(t_i))z_1 z_0. \end{aligned} \tag{38}$$

2. Generic expression of G_{q_1,t_i} up to order $k = 2$:

$$\begin{aligned} G_{q_1,t_i} = q_1(t_i) &+ sin(q_2(t_i))z_0 + \\ &sin(q_3(t_i))cos(q_2(t_i))z_0^2. \end{aligned} \tag{39}$$

3. Generic expression of G_{q_3,t_i} up to $k = 2$:

$$\begin{aligned} G_{q_3,t_i} = q_3(t_i) &+ (q_4(t_i))^3 z_0 + z_1 + \\ &3(-(q_4(t_i))^3 - (q_1(t_i))^{10})(q_4(t_i))^2 z_0^2. \end{aligned} \tag{40}$$

4. Generic expression of G_{q_4,t_i} up to $k = 2$:

$$\begin{aligned} G_{q_4,t_i} = q_4(t_i) &+ (-(q_4(t_i)^3 - (q_1(t_i))^{10})z_0 + \\ &(3((q_4(t_i))^3 + (q_1(t_i)^{10}))(q_4(t_i))^2 \\ &-10(q_1(t_i))^9 sin(q_2(t_i)))z_0^2. \end{aligned} \tag{41}$$

5. Generic expression of $Y_i(h) = q_2(t_i + h)$ up to $k = 2$:

$$\begin{aligned} Y_i(h) &= y(t_i + h) \\ &= y(t_i) + \langle G_{y,t_i} | z_0 \rangle h + \\ &\quad \langle G_{y,t_i} | z_0^2 \rangle h^2/2 + \langle G_{y,t_i} | z_1 z_0 \rangle (\psi_{i,2}(h)). \end{aligned} \tag{42}$$

6. Generic expression of $q_1(t_i+h)$ up to $k=2$:

$$\begin{aligned} q_1(t_i+h) = q_1(t_i) &+ \langle G_{q_1,t_i}|z_0\rangle h + \\ &\langle G_{q_1,t_i}|z_0^2\rangle h^2/2. \end{aligned} \tag{43}$$

7. Generic expression of $q_3(t_i+h)$ up to $k=2$:

$$\begin{aligned} q_3(t_i+h) = q_3(t_i) &+ \langle G_{q_3,t_i}|z_0\rangle h + \\ &\langle G_{q_3,t_i}|z_1\rangle(\psi_{i,1}(h)) + \\ &\langle G_{q_3,t_i}|z_0^2\rangle h^2/2. \end{aligned} \tag{44}$$

8. Generic expression of $q_4(t_i+h)$ up to $k=2$:

$$\begin{aligned} q_4(t_i+h) = q_4(t_i) &+ \langle G_{q_4,t_i}|z_0\rangle h + \\ &\langle G_{q_4,t_i}|z_0^2\rangle h^2/2. \end{aligned} \tag{45}$$

4.5 Maple Package: Some Demonstrations

In this section, we produce a demonstration in the following cases:

– For stable system (electric equation with positive parameters) for oscillating input $u(t) = sin(100t)$, step=0.01 (Runge-Kutta vs our method). The drawings are similar by both methods. See Fig. 1

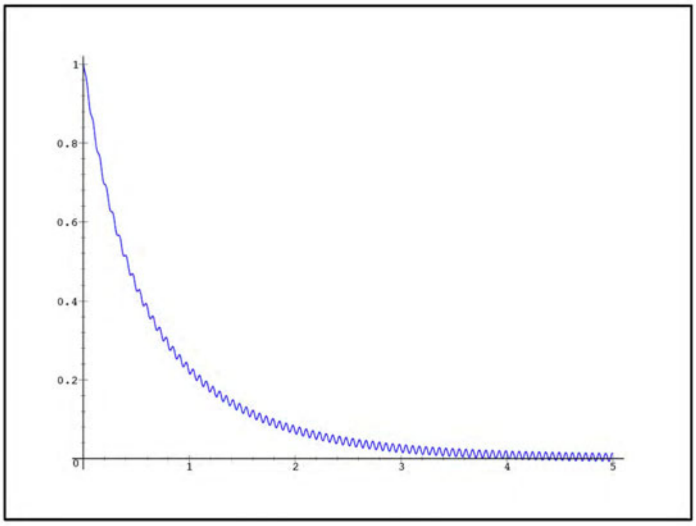


Fig. 1. Stable system, oscillating input, small step, by Runge-Kutta or our method

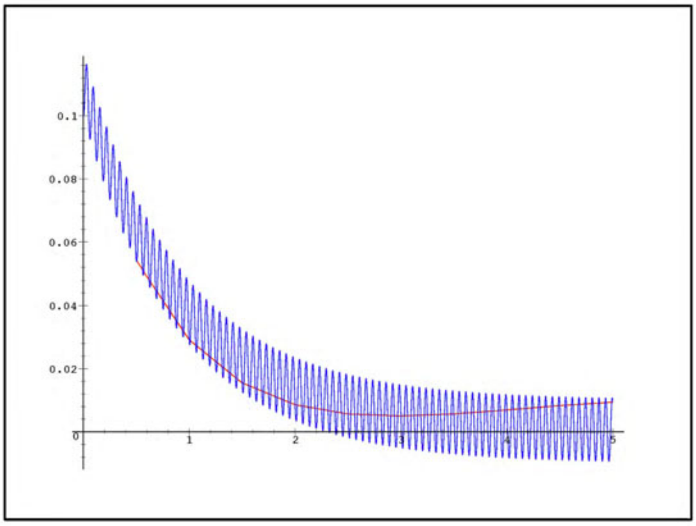


Fig. 2. Stable system, oscillating input, large step, by Runge-Kutta (without oscillation) and our method

– For stable system (electric equation, linear equation) with oscillating output (Runge-Kutta vs our method vs Exact Solution)

1. Electric equation with positive parameters for oscillating input $u(t) = sin(100t)$, step= 0.5 (Runge-Kutta vs our method). The oscillations of the output are not described by Runge-Kutta method when our method diplays a lot of oscillations. See Fig. 2.
2. Linear equation for oscillating input $u(t) = t^2 sin(100t)$, step= 0.05 (Runge-Kutta vs our method vs Exact Solution). The drawing of the exact solution is similar to the drawing of our method. See Figs. 3,4,5.
3. Dynamical system (Example 2) for oscillating input $u(t) = t^2 sin(10t)$, step= 0.5 (Our method at order 2 vs Exact Solution). The drawing of the exact solution is exactly the drawing of our method at order 2 (identical curves). See Fig. 6.

– For unstable system (electric equation with negative parameters), $u(t) = sin(100t)$, step = 0.01., Runge-Kutta method notifies an error when our method displays a suitable curve. The Runge-Kutta method does not apply to this case when the drawing of our method displays an infinite branch. See Fig. 7.

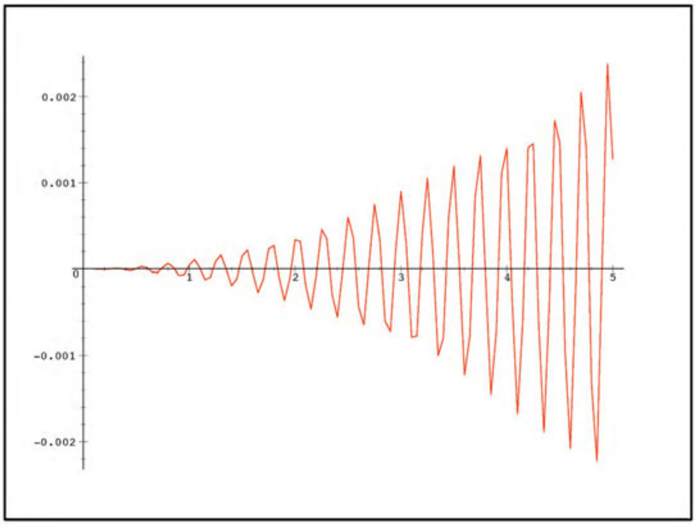


Fig. 3. Linear equation, oscillating input, small step by Runge-Kutta

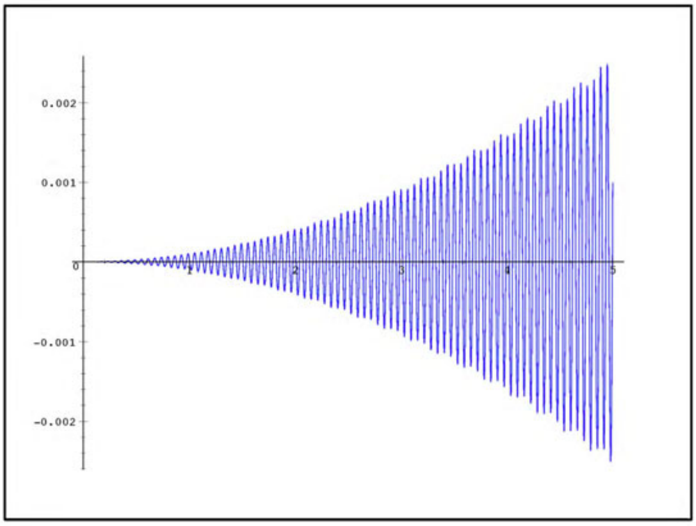


Fig. 4. Linear equation, oscillating input, small step by our method

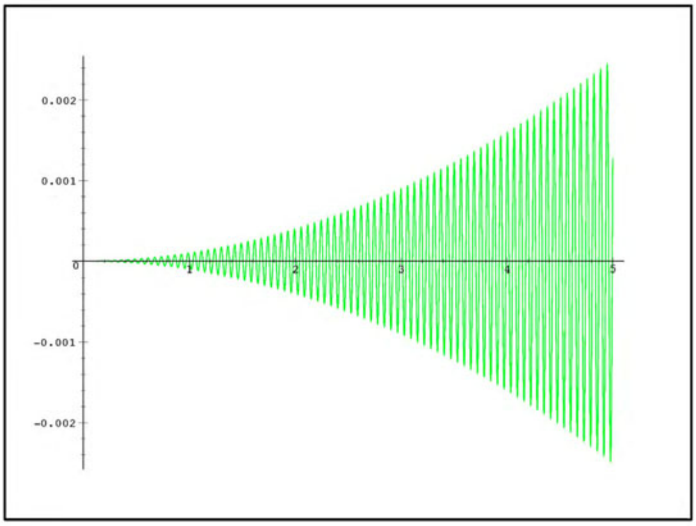


Fig. 5. Linear equation, oscillating input, small step by exact method

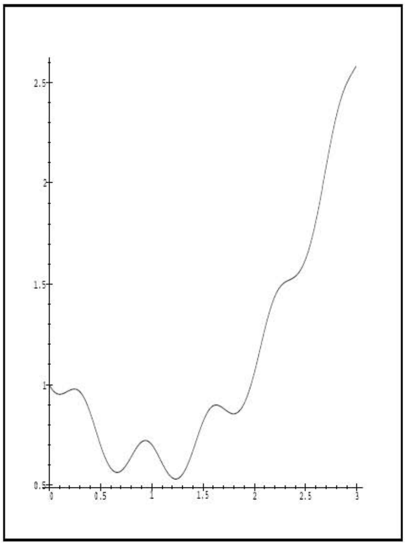


Fig. 6. Dynamical system, oscillating input, by exact method and our method

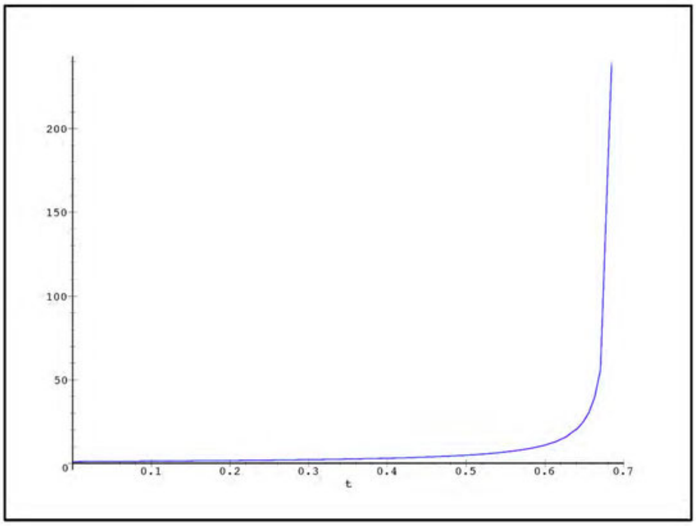


Fig. 7. Unstable system, oscillating input, small step, by our method

5 Some Practical Applications

In biology, a lot of phenomenons are oscillating and sometimes cyclic. For instance, the oscillations of the hormonal secretions present some periodic activities. In the cellular cycle, the synchronization of the oscillations is basic.
An interesting application of our method in pharmacodynamy consists in modeling the effect of a drug on the organism in terms of time, in predicting the behaviour of the organism and in regulating the parameters of the organism by distributing suitably the drug in terms of time. This work is based on the associated drawings (in term of time) of the values of the drug and of the values of the studied parameter.

In diabetology, by infusing insulin to diabetic patients, and by measuring the corresponding glycaemia, we obtain two associated curves (insulin delivery and glycaemia rate in the blood, in terms of time). With a sufficient set of these pairs, we can provide a modeling in the form of a dynamical system [11]. And then the prediction, regulation [8] can be processed by the knowledge of an accurate drawing of the model.

The detection of climate cycles in climatology is studied and several models are provided. The prediction of these cycles would be perhaps processed by the knowledge of an accurate drawing of a suitable model.

6 Conclusion

We develop a method for drawing the output of a dynamical system, based on the symbolic computing.

The symbolic computing allows us to profit from the genericity: We propose that one uses the formal expression of the generating series $G_{y,t}, G_{q_r,t}$, of the output $y(t)$ and the components of the state $(q_r)_{1\leq r\leq N}$. Then we replace these expressions by their values at every step.

The symbolic computing allows us to profit from the precision: We can choose any order k for approximating the output. The error is on the order of $k+1$.

And then an interest of this method consists in choosing the precision, not only by the size of the time interval h but by the order of the approximation. The quality of any approximation depends on the order, the size of the interval but also depends on the roughness of the curve and the stability of the system. From a lot of examples, we express the following conclusions:

For stable systems with smooth outputs, our method and a Runge-Kutta method provide similar results.

For unstable systems, our methods allows us to obtain a suitable result near the instability points, when the Runge-Kutta methods give an error message. For stable systems with rough or oscillating outputs, our method provides a suitable result when a Runge-Kutta method is wrong. Many applications in physics or biology are then available. The detection of climate cycles in climatology, the study of the synchronization of oscillations in biology, some problems of pharmacodynamy (modeling, prediction, regulation) would be partly processed by the knowledge of an accurate drawing of the model.

Acknowledgments. The authors are grateful to Michel Petitot for giving us some ideas for dealing with dynamical systems instead of differential equations.

References

1. Akhrif, O., Blankenship, G.L.: Algebraic computations for design of nonlinear control systems. In: Jacob, G., Lamnabhi-Lagarrigue, F. (eds.) Algebraic Computing in Control. LNCIS, vol. 165, pp. 129–155. Springer, Heidelberg (1991)
2. Benmakrouha, F., Hespel, C., Monnier, E.: Drawing solution curve of a differential equation. In: 3rd International Conference on Complex Systems and Applications (ICCSA 2009), Le Havre (2009)
3. Chen, K.T.: Iterated path integrals. Bull. Amer. Math. Soc. 83, 831–879 (1977)
4. Fliess, M.: Fonctionnelles causales non linéaires et indéterminés non commutatives. Bull. Soc. Math. France 109, 3–40 (1981)
5. Fliess, M., Reutenauer, C.: Théorie de Picard-Vessiot des systèmes réguliers. Colloque Nat.CNRS-RCP 567, Outils et Modèles Mathématiques pour l'Automatique, l'Analyse des Systèmes et le Traitement du Signal, Belle-Ile (September 1982)
6. Fliess, M., Lamnabhi, M., Lamnabhi-Lagarrigue, F.: An algebraic approach to nonlinear functional expansions. IEEE Trans. Circuits and Systems CAS-30, 554–570 (1983)

7. Foursov, M.V., Hespel, C.: On formal invertibility of the input/output behavior of dynamical systems. Publication interne IRISA, N° 1439 (2002)
8. Foursov, M.V., Hespel, C.: On Algebraic Modeling and regulation of the behavior of diabetics. In: Innovative Technologies for Insulin Delivery and Glucose Sensing. Aix en Provence (August 2003)
9. Hespel, C.: Iterated derivatives of a nonlinear dynamic system and Faà di Bruno formula. Mathematics and Computers in Simulation 42, 641–657 (1996)
10. Hespel, C.: Une étude des séries formelles non commutatives pour l'Approximation et l'Identification des systèmes dynamiques. Thèse d'état, Université de Lille 1 (1998)
11. Hespel, C., Hespel, J.P., Monnier, E., Jacob, G., Foursov, M.V., Benmakrouha, F.: Algebraic Identification: application to insulin infusion. In: ISGIID 2000, Evian (September 2000)

Practical Mixed-Integer Optimization for Geometry Processing

David Bommes, Henrik Zimmer, and Leif Kobbelt

RWTH Aachen University,
Ahornstr. 55, 52074 Aachen, Germany
http://graphics.rwth-aachen.de

Abstract. Solving mixed-integer problems, i.e., optimization problems where some of the unknowns are continuous while others are discrete, is NP-hard. Unfortunately, real-world problems like e.g., quadrangular remeshing usually have a large number of unknowns such that exact methods become unfeasible. In this article we present a greedy strategy to rapidly approximate the solution of large quadratic mixed-integer problems within a practically sufficient accuracy. The algorithm, which is freely available as an open source library implemented in C++, determines the values of the discrete variables by successively solving relaxed problems. Additionally the specification of arbitrary linear equality constraints which typically arise as side conditions of the optimization problem is possible. The performance of the base algorithm is strongly improved by two novel extensions which are (1) simultaneously estimating sets of discrete variables which do not interfere and (2) a fill-in reducing reordering of the constraints. Exemplarily the solver is applied to the problem of quadrilateral surface remeshing, enabling a great flexibility by supporting different types of user guidance within a real-time modeling framework for input surfaces of moderate complexity.

Keywords: Mixed-Integer Optimization, Constrained Optimization.

1 Introduction

The problem of optimizing objective functions $E(\mathbf{x})$ where one part of the unknowns is real valued $x_{i\in R} \in \mathbb{R}$ and the other unknowns are required to be integers $x_{j\in I} \in \mathbb{Z}$ is referred to as *Mixed-Integer Problem* (MIP). Typically such problems arise whenever a continuous optimization depends on some discrete decisions. One example from structural engineering is the optimization of a supporting structure, where two points of this structure can either be connected by a beam or not while in contrast to this discrete decision the geometric positions of the connected points can be varied continuously.

Mixed-Integer Problems are generally NP-hard to solve [1,2], which make common approaches (e.g., Branch and Bound [3,4,2] or Cutting-Plane approaches [5,2]) unfeasible for many real-world problem instances.

In this article we present an algorithm for efficiently and accurately approximating *quadratic* MIPs represented by quadratic energy functions

$$E(\mathbf{x}) = \frac{1}{2}\mathbf{x}^t A\mathbf{x} - \mathbf{x}^t\mathbf{b} \rightarrow min, \quad \mathbf{x} \in \mathbb{R}^n \tag{1}$$

with A symmetric and positive definite, subject to n_I integer constraints

$$x_{i\in I} \in \mathbb{Z}, \quad I \subseteq \{1,\ldots,n\}. \tag{2}$$

Additionally the feasibility of the solution $\mathbf{x}$ is restricted by n_C linear *equality* constraints of the form

$$\mathbf{C}_i \cdot \mathbf{x} = d_i \qquad \text{with} \qquad \mathbf{C}_i \in \mathbb{R}^n,\ d_i \in \mathbb{R} \tag{3}$$

which can be assembled into a single matrix $C\mathbf{x} = \mathbf{d}$ with dimension $C \in \mathbb{R}^{n_C \times n}$. Here n, n_I and n_C denote the number of variables, integer constraints and linear constraints respectively. Note also that the above formulation differs slightly from the most general setting of mixed-integer problems where additionally *in-equality* constraints are given. In the presence of in-equality constraints even finding any random feasible solution can get very hard, see e.g., [2].

Our algorithm successively determines the values of the discrete variables $x_{i\in I} \in \mathbb{Z}$ in a greedy fashion. Fixing the value of a discrete variable is equivalent to adding one explicit linear constraint $x_i = k$ with $k \in \mathbb{Z}$ to our optimization problem. Therefore our algorithm successively transforms integer constraints into explicit linear constraints until all of them are fulfilled. More precisely we start by neglecting the n_I integer constraints and compute the minimizer of this so called relaxed problem by setting the partial derivatives $\frac{\partial E}{\partial x_i} = 0$ and solving the resulting linear system

$$A\mathbf{x}^0 = \mathbf{b}. \tag{4}$$

Here we assumed $n_C = 0$ for clarity reasons. The values of the solution vector $\mathbf{x}^0$ can be seen as continuous estimates of the desired discrete integer variables. However, we found that estimating all integer constraints at once, i.e., requiring $\forall i \in I : \ x_i^1 = round(x_i^0)$, leads to poor results since the individual estimates cannot influence each other. Motivated by this observation we instead successively determine single integer constraints $x_j^{k+1} = round(x_j^k)$ which are henceforth used to solve subsequent relaxed problems until a feasible solution of our initial optimization problem is found, meaning that all $x_{i\in I}$ are integers.

By greedily choosing the continuous estimate which has the smallest deviation $|round(x_j^k) - x_j^k|$ from an integer in each step these subsequent relaxed problems can be solved very efficiently by a carefully designed three-level solver as presented in Section 3.2. The performance can further be improved by identifying sets of relaxed variables which do not interfere too much and hence can be estimated simultaneously.

In order to facilitate an efficient handling of arbitrary linear constraints C_i we propose to eliminate one variable for each constraint (Section 2) and support this approach by a fill-in reducing constraint reordering (Section 2.2) which in practice significantly reduces the runtime.

In Section 4 the capabilities of the presented solver are illustrated exemplary by applying it to the surface quadrangulation problem. A large variety of possible user guidance like e.g., prescribing some singularities or preserving feature curves of the input geometry in the generated quadrangulation is achieved by simply adding additional linear constraints. Therefore the presented algorithm enables a powerful quadrangulation algorithm which is very flexible and allows for a wide range of different application scenarios ranging from a fully automatic setting up to a complete manual meshing. Finally we illustrate the immense performance benefit due to the novel extensions of the original algorithm used in [6], i.e., the rounding of sets of variables and the fill-in reducing reordering of constraints.

1.1 Previous Work

To the best of our knowledge the idea of approximating MIPs by a series of real-valued problems started with [13], set in the field of Structural Engineering. Ringertz' idea of rounding variables iteratively and re-solving the problem has been cited several times and depending on the problem setting small variations appear in the proposed solutions. While some researchers argue for the use of post-processing methods as, depending on the problem and the type of variables, always rounding up (or down) might not be meaningful [16], others suggest rounding both up and down and keeping the solution with lower cost [17].

Regardless of the rounding strategy, what these approaches all have in common is that the full-sized system of linear equations needs to be re-solved in each iteration, making the iterative strategy unfeasible for many practical applications.

In the field of Geometry Processing the idea of approximating quadratic MIPs by rounding variables of a real-valued linear system has been successfully adapted by several authors (see e.g.,[9][11]). Here direct-rounding strategies were used, where the system had to be solved only twice, once initially and once after all integer constraints have been estimated (all at once). This approach is usually only applicable for MIPs with a small number of integer variables that do not interfere too much and otherwise leads to a poor approximation of the optimal solution.

The article is structured as follows: Section 2 describes the proper handling of linear constraints within the optimization of a quadratic energy, which is central also for the integer constraints discussed in Section 3. In Section 3.2 we describe an efficient update strategy which enables the iterative addition of integer constraints. Finally in Section 4 we discuss some experiments performed within the context of quadrilateral surface remeshing.

2 Linear Constraints

The ability to properly handle constraints is vital for the wide applicability of an optimization method. For a problem to be solvable usually some boundary constraints are needed to limit the solution space, or often additional user-defined design constraints might be incorporated to shape the resulting solution. In our setting we also have to deal with integer constraints which translate into simple linear conditions as soon as the specific integer is known. A common way to handle linear constraints are *Lagrangian Multipliers* as discussed next.

2.1 Lagrangian Multipliers

The method of Lagrangian Multipliers turns a constrained problem into a unconstrained one by adding one additional term per constraint to the energy. Updating energy (1) with the constraints (3) we end up with the following energy formulation:

$$E_L(\mathbf{x}) = E(\mathbf{x}) + \sum_{i=1}^{n_C} \lambda_i (C_i \cdot \mathbf{x} - d_i) \tag{5}$$

where the solution is given by the following system of linear equations

$$\frac{\partial E_L}{\partial \boldsymbol{x}} = 0 \quad \rightsquigarrow \quad \begin{bmatrix} A & C^T \\ C & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{d} \end{bmatrix} \tag{6}$$

describing the stationary point of the adapted energy. Note that the approach of Lagrangian Multipliers is not restricted to quadratic energies nor linear constraints but can be applied to non-linear problems as well, for more details see e.g., [10]. Unfortunately the approach of Lagrangian Multipliers comes with certain disadvantages making them impractical for our purpose. Instead of decreasing the number of degrees of freedom as more constraints are added the opposite is the case since for each constraint a Lagrangian Multiplier λ_i is introduced. Furthermore the symmetric positive definiteness (s.p.d.), inherent in linear systems arising from convex quadratic energies is destroyed by the diagonal block of zeros $\mathbf{0}$ effectively disabling the use of highly efficient solvers such as CHOLMOD (see [7]) and necessitating the use of slower more general solvers such as SuperLU (see [8]). As will be seen in Section 3 the s.p.d. property is crucial for the efficient local updates of the adaptive three-level solver in Section 3.2. Therefore next we describe a proper handling of linear constraints which maintains the s.p.d. property.

2.2 Elimination Approach

Assume we want to minimize a quadratic energy $E(\mathbf{x})$ with $\mathbf{x} \in \mathbb{R}^n$ subject to a single linear constraint $\mathbf{D}^T\mathbf{x} - d = 0$. Geometrically this means restricting the solution space to a $n-1$ dimensional hyperplane. Consequently it is possible to convert the above problem into a new unconstrained one with $n-1$ degrees of freedom. Assume w.l.o.g. that $D_n \neq 0$ such that we can solve the linear constraint for x_n expressing it as a function of $(x_1, \ldots, x_{n-1})$

$$x_n(\underbrace{x_1, \ldots, x_{n-1}}_{\tilde{\mathbf{x}}}) = (d/D_n) - \sum_{j=1}^{n-1} (D_j/D_n) x_j =: f - \mathbf{F}^T\mathbf{x} \tag{7}$$

and transforming the above constrained problem into the desired unconstrained form

$$\tilde{E}(\tilde{\mathbf{x}}) := E \underbrace{\begin{pmatrix} \tilde{\mathbf{x}} \\ x_n(\tilde{\mathbf{x}}) \end{pmatrix}}_{\mathbf{y}} \tag{8}$$

with equivalent minima where x_n can be computed from $\tilde{\mathbf{x}}$ by equation (7).

To compute the minimizer $\tilde{\mathbf{x}}$ we now have to solve a $(n-1)$ dimensional system of linear equations $\tilde{A}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}$ which can be derived by partitioning the matrix A of equation (1) into four blocks (with $\overline{A} \in \mathbb{R}^{(n-1)\times(n-1)}$, $\boldsymbol{v} \in \mathbb{R}^{n-1}$ and $w \in \mathbb{R}$) and re-factorizing the result:

$$\begin{aligned} \tilde{E}(\tilde{\mathbf{x}}) &= \frac{1}{2}\mathbf{y}^T A \mathbf{y} - \mathbf{y}^T \mathbf{b} \;=\; \frac{1}{2}\mathbf{y}^T \begin{pmatrix} \overline{A} & \mathbf{v} \\ \mathbf{v}^T & w \end{pmatrix} \mathbf{y} - \mathbf{y}^T \begin{pmatrix} \overline{\mathbf{b}} \\ b_n \end{pmatrix} \\ &= \frac{1}{2}\tilde{\mathbf{x}}^T \underbrace{\left(\overline{A} - \mathbf{v}\mathbf{F}^T - \mathbf{F}\mathbf{v}^T + \mathbf{F}\mathbf{F}^T\right)}_{\tilde{A} \,\in\, \mathbb{R}^{(n-1)\times(n-1)}} \tilde{\mathbf{x}} - \tilde{\mathbf{x}}^T \underbrace{\left(\overline{\mathbf{b}} + \mathbf{F}(fw + b_n) - \mathbf{v}f\right)}_{\tilde{\mathbf{b}} \,\in\, \mathbb{R}^{n-1}} + \text{const} \end{aligned} \tag{9}$$

Note that since A is s.p.d. $\tilde{A}$ is also s.p.d. enabling highly efficient solution methods used in our three-level solver described in Section 3.2. Of course instead of eliminating the last variable each other variable can be chosen by re-indexing.

Multiple Constraints: In general we want to handle an arbitrary number of linear constraints which can be achieved by iteratively eliminating one variable for each constraint from $\{C_1, \ldots, C_{n_C}\}$. One very important aspect of multiple constraints is that in each step it is necessary to eliminate the chosen variable from all subsequent constraints since obviously once a variable is constrained and eliminated from the optimization problem it should not be reintroduced by a following constraint. More precisely, after constraining a variable x_k through a constraint C_j we have to do Gaussian elimination in the constraint matrix C in order to bring all $C_{i,k}$ with $i > j$ to zero. Clearly the constraints in the updated matrix are equivalent to the original problem.

Choosing elimination variables: For each linear constraint we have to pick a variable which is subsequently constrained by the induced linear function and eliminated from the problem. All non-zero coefficients of the linear constraint induce a valid possibility. To increase numerical stability we select the variable whose coefficient has the largest absolute value. However, there is one important aspect to consider whenever a variable x_k with $k \in I$, i.e., which has to satisfy an integer constraint, is selected for elimination. In general it can get very problematic to guarantee that the values of the induced linear function are chosen in such a way that x_k becomes an integer. Consequently for the elimination we always prefer non-integer variables with $k \notin I$. For constraints where all non-zero coefficients belong to integer variables we currently support only those cases where all coefficients are integer and their greatest common divisor (gdc) is one of these coefficients. In such a case we can safely divide all coefficients by their gdc and eliminate a variable with coefficient ± 1 since a linear combination of integers multiplied by integers is always an integer and consequently the integer constraint is fulfilled by construction. For many practical problems (like quadrilateral surface remeshing) the above assumptions are always fulfilled and therefore we leave the more complicated general case for future work. Whenever the above assumption is violated it may happen that in the result some of the integer constraints are not fulfilled.

Linear dependent or conflicting constraints: Since we iteratively process the individual constraints it is easy to identify linear dependent or conflicting constraints. This is a big advantage compared to the method of Lagrangian Multipliers which would construct an underdetermined system of linear equations not suitable for efficient standard solvers. In our implementation linear dependent or conflicting constraints are simply neglected. This behavior is very convenient since the user does not have to spend additional effort identifying the subset of linear independent constraints e.g., in the case of user provided side conditions. Due to numerical inaccuracies of floating point numbers linear dependency is checked against a tolerance which has a default value of 10^{-6}.

Fill-in reducing constraint reordering: Although mathematically equivalent the linear system belonging to the unconstrained optimization problem after processing all constraints can take many different patterns, strongly depending on the processing sequence of the constraints. In spirit of sparse Cholesky methods like [7] we are interested in finding an ordering of the constraints which minimizes the fill-in (nonzero elements) and hence increases performance. Unfortunately there is no known algorithm to achieve the best ordering apart from the naive one which explicitly checks all orderings. Obviously such an approach is far too slow such that a good compromise in form of a cheap heuristic is more desirable. Our experiments show that processing the constraints sorted by their number of non-zero coefficients leads to much higher performance than just using a random ordering (see Section 4 for timings). Please note that this ordering is dynamic since while processing the constraints the number of non-zero coefficients is altered by the Gaussian elimination steps.

After eliminating one variable per linear constraint we obtain a new equivalent optimization problem, i.e., a quadratic energy minimization subject to a set of integer constraints. We describe next how a good approximate solution can be found efficiently.

3 Integer Constraints

The integer constraints of our initial problem dictate that for each feasible solution a subset of the variables have to be integers, i.e., $x_{i\in I} \in \mathbb{Z}$. Finding a feasible solution is simple in this formulation, since there are no dependencies between the individual variables. Therefore just setting up a set of additional linear constraints which fix the $x_{i\in I}$ variables to arbitrary integers like e.g., $x_{i\in I} = 0$ and

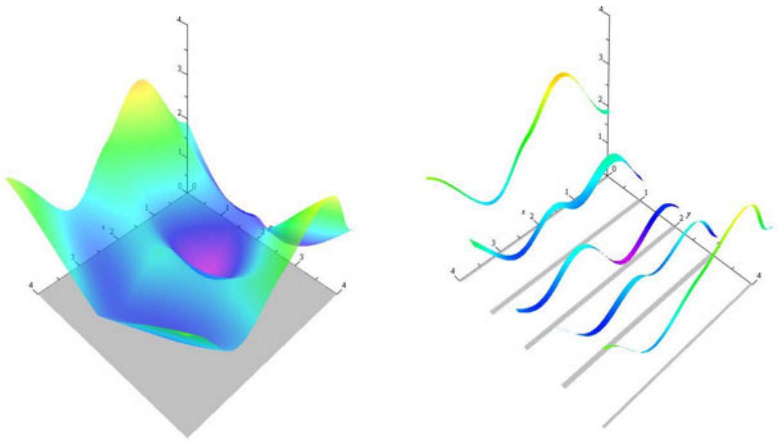


Fig. 1. (left) a continuous optimization problem where each point in the plane $\mathbb{R} \times \mathbb{R}$ is a feasible solution, i.e., a point which fulfills all constraints of the problem. (right) a mixed-integer problem where the set of feasible solutions is $\mathbb{R} \times \mathbb{Z}$. For minimizing such problems typically all discrete possibilities have to be tested explicitly.

enforcing them with the method from the previous section would indeed result in a feasible solution. However the problem of finding the best one of all these possible assignments, i.e., the one which minimizes the energy, is very hard. In contrast to continuous convex optimization it is not sufficient to simply walk into the direction of the negative gradient (see Figure 1). In general to find the optimum it would be necessary to derive lower and upper bounds for each discrete variable and then explicitly test all discrete combinations. Please notice that for problems with a large number of discrete variables even for narrow bounds like e.g., a binary problem where $x_{i \in I} \in \{0, 1\}$ such an approach is very expensive and would already require the solution of $2^{|I|}$ full-sized problems.

3.1 Direct Rounding

Instead of achieving optimality for practical problems we aim at finding an approximate solution which is close enough to the optimum but can be computed in a fraction of time. The most efficient way to determine adequate assignments for the integer variables is to estimate them from a relaxed solution, i.e., computing the minimizer $\boldsymbol{x}^c$ where all variables are allowed to be continuous leading to the estimates $x_{i \in I} = round(x_i^c)$. Following (9) the elimination approach results in a very simple update for such explicit constraints:

$$\tilde{A} = \overline{A} \quad \text{and} \quad \tilde{\boldsymbol{b}} = \overline{\boldsymbol{b}} - \boldsymbol{v} \cdot round(x_i^c) \tag{10}$$

Estimating all integer assignments at once which is called *direct rounding* is very efficient since it requires the solution of only two linear systems. However the drawback is that the interrelation between the discrete variables is completely ignored which often leads to poor results (see e.g., the comparison in [6]). This suggests to successively add one integer constraint at a time and immediately compute the altered relaxed problem to update the estimates of the yet unconstrained discrete variables. This strategy is denoted *iterative rounding* and is discussed in more detail next.

3.2 Iterative Greedy Rounding

The key to an efficient implementation of the iterative rounding is the observation that, for problems with sparse variable dependencies (few non-zeros per row), changing the value of one variable usually has little influence on "far-away" variables. This is a property inherent in many Geometry Processing problems formulated over, e.g., simplicial complexes or spline bases with local support.

The problem inherent to iterative rounding is that it requires the solution of $|I|+1$ many linear systems which can get very slow when implemented in a naïve way. Fortunately in many steps of this iterative process the solution changes only slightly which can be exploited by carefully designed iterative solvers.

Suppose that we have computed the solution of the relaxed problem $A\boldsymbol{x} = \boldsymbol{b}$ and that we want to add a single integer constraint. Following (10) the residual $\boldsymbol{e} = \tilde{A}\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{b}}$ after adding the new constraint has the same nonzero pattern as $\boldsymbol{v}$. And consequently for a sparse $\boldsymbol{v}$ the relaxed solution from the previous step $\tilde{\boldsymbol{x}}$ violates only a few equations of the linear system. Due to this observation we first try to iteratively update the solution only where it is necessary, i.e., for all variables $\tilde{x}_i$ with $|e_i| > \epsilon$. This so called *Local Gauss-Seidel* method executes single Gauss-Seidel updates for variables with a local residual above the allowed tolerance. All these candidates are stored in a queue and convergence is reached when the queue gets empty meaning that all residuals are below the prescribed tolerance. Notice that due to the elimination approach the system matrix remains s.p.d. guaranteeing convergence of the Gauss-Seidel method. The complete algorithm is depicted below:

```
Algorithm: Local Gauss-Seidel
Input: Linear system $A\boldsymbol{x} = \boldsymbol{b}$ (which is not fulfilled)
       Index set of variables with non-zero residual $N$,
       End conditions $\epsilon$ and $maxiters_{GS}$
Output: Updated $\mathbf{x}$ with residuals $|e_k| < \epsilon$ or NOT converged.
01: push $N$ onto queue
02: $iter = 0$
03: while queue not empty and $iter < maxiters_{GS}$
04:     iter = iter +1
05:     $x_k$ = pop( queue )
06:     $e_k = b_k - \sum_{j=1}^{n} A_{kj} x_j$
07:     if $|e_k| > \epsilon$ then
```

```
08:         $x_k \leftarrow x_k + e_k/A_{kk}$
09:         push $nonzero(A_{k*})$ onto $queue$
07:     **end if**
10: **end while**
```

The parameters ϵ and $maxiters_{GS}$ can be chosen by the user. In cases where the above method does not converge within the prescribed number of iterations, a more global conjugate gradient method is used and in rare cases where this is still not sufficient after a few iterations a sparse Cholesky method is executed. This adaptive solution strategy is very fast if the previous solution is close to the new one and only spends more time if a novel integer constraint has global impact. In our implementation the conjugate gradient solver is taken from the GMM++ library [12] and the Sparse Cholesky solver is the CHOLMOD solver [7].

In this iterative rounding strategy we can choose $|I|!$ many different orders in which the integers are estimated. A natural greedy choice is the yet unconstrained integer variable whose estimate has the smallest deviation $|x_i - round(x_i)|$ from an integer since it is most likely to be correct. A nice side effect of this strategy is that it increases the efficiency of the above hierarchical solution strategy. The reason is that for small deviations from an integer also the non-zero residuals usually get small. The complete iterative greedy rounding algorithm is shown below:

```
Algorithm: *Iterative Greedy Rounding*
**Input:** Linear system of relaxed problem $A\boldsymbol{x}^c = \boldsymbol{b}$ with $\boldsymbol{x}^c, \boldsymbol{b} \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$
        index set of integer variables $I \subset \{1, \dots, n\}$
**Output:** Approximation of mixed-integer solution $\boldsymbol{x} \in \mathbb{R}^n$ satisfying $x_{i\in I} \in \mathbb{Z}$
01: $\boldsymbol{x} = \boldsymbol{x}^c$
02: **while** $I \neq \emptyset$
03:     *// greedy selection*
04:     $j = \arg\min_{i\in I}(|x_i - round(x_i)|)$
05:     $I \leftarrow I \setminus j$
06:     *// add new constraint and get nonzero residuals N*
07:     $N$ = eliminateConstraint( $x_j = round(x_j)$ , $A$ , $\boldsymbol{x}$ , $\boldsymbol{b}$ )
08:     *// update solution*
09:     converged = localGaussSeidel( $A$, $\boldsymbol{x}$, $\boldsymbol{b}$, $N$) *// level 1*
10:     **if** not converged **then**
11:         converged = conjugateGradient( $A$, $\boldsymbol{x}$, $\boldsymbol{b}$) *// level 2*
14:         **if** not converged **then**
15:             sparseCholesky( $A$, $\boldsymbol{x}$, $\boldsymbol{b}$) *// level 3*
16:         **end if**
17:     **end if**
18: **end while**
```

To avoid the necessary re-indexing of the variables in the above algorithm the update rule (10) was slightly modified by keeping an identity row and column for each eliminated variable x_k, i.e., $\tilde{A}_{kj} = \tilde{A}_{jk} = \delta_{kj} \ \ \forall j$.

In our implementation the user is able to control the behavior of the adaptive three level solver with several parameters. First of all the tolerance ϵ for checking convergence of the iterative methods (level 1 and 2) and a maximum number of iterations $maxiters_{GS}$ and $maxiters_{CG}$ can be adjusted. Furthermore it is possible to disable complete levels. The reason is that for mixed-integer problems where it is known that the rounding of a discrete variable always has global impact it is e.g., not reasonable to execute the Local Gauss-Seidel step since it would almost never converge. Therefore it is very important to experiment a little bit with these parameters in order to optimize the performance for a specific class of problems. In Section 4 we will provide two different useful settings for the quadrangulation problem.

Simultaneous Rounding: The motivation for the iterative rounding strategy was mainly the observation that the estimates of individual integer variables should influence each other to achieve satisfactory accuracy. It would be possible to achieve the same accuracy in fewer computation steps if some prior knowledge about the rate of influence between variables is available. Clearly variables which do not influence each other could be rounded simultaneously in one step without introducing an error. Unfortunately computing the influence between variables corresponds to the solution of a full-sized linear system which would be too expensive. What we need instead is a cheap apriori estimate which never underestimates the interdependency. A very simple apriori estimate which holds for many problems is the following one: If one variable is changed by a value of Δx due to a constraint all other variables are changed by a value smaller or equal than Δx. Consequently in each step several variables can be rounded as long as their estimated maximal deviation $\sum_i \Delta x_i$ does not influence any of the rounding decisions. Obviously this apriori estimate does not hold for all problems. However we included the possibility to use it into our implementation since it is useful for many practical applications and can speed up the computation significantly. Finding a cheap way for estimating sharper bounds for the interdependency between discrete variables is an interesting question for future work.

3.3 Open Source CoMISo Library

On the web page `http://www.graphics.rwth-aachen.de/comiso` the source-code of the solver explained above as well as example programs can be found. Note that even though this solver was created for sparse problems as they usually occur e.g., in areas of Finite Elements or Geometry Processing, it can also be applied to dense problems without any modifications. However, in some cases it might be advantageous to replace sparse-specific parts such as the sparse Cholesky solver by dense-optimized counterparts.

4 Experiments

We evaluate our algorithm by applying it to the surface quadrangulation problem as formulated in [6]. In this method two mixed-integer problems have to be

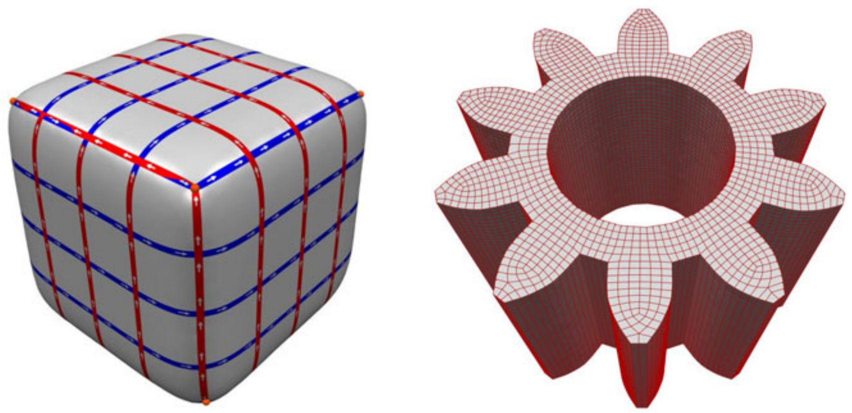


Fig. 2. (left) the SMOOTHED CUBE model with low geometric complexity. (right) the PINION model with many sharp features.

solved where the first one is the computation of a smooth orientation field while the second one is a seamless parametrization mapping singularities and feature edges to integer grid points and lines respectively. For more details about the quadrangulation method we refer the reader to [6]. With the help of several experiments we derived two different parameters for the two diverse problems. For the computation of the orientation field we used $\epsilon = 10^{-3}$, $maxiters_{GS} = 100000$ and $maxiters_{CG} = 50$ while for the parametrization we chose $maxiters_{GS} = 0$, $maxiters_{CG} = 20$ and the use of sparse cholesky was disabled completely. The reason for those different parameter settings is that both problems have very diverse characteristics. While the orientation field exhibits a large number of integers with local influence, the parametrization problem requires only few integers but with rather global influence. With the above settings we were able to compute visually equivalent results compared to the original algorithm of [6] within a fraction of time. The performance benefit is a result of the tuned parameters as well as the novel extension which are the fill-in reducing reordering, the simultaneously rounding and some changes within the internal data structures. All examples were computed on a single CPU of an intel i7 quadcore 2.80GHz with 8GB of RAM.

Performance: To give one representative example the orientation field computation on the LEVER model of [6] took $3.3s$ compared to$0.22s$ while the parametrization timing decreases from $19.9s$ to $2.8s$. However, further experiments showed that the runtime strongly depends on the geometric complexity of the object. In Table 1 we compare the timing of the orientation field computation of the ARMADILLO model (Figure 3) and a simple SMOOTHED CUBE (Figure 2). For the same number of triangles the geometric more complex ARMADILLO (121 singularities) model needs more computation time than the smoothed cube (4 singularities). In the case of constant geometric complexity the runtime depends

Table 1. Orientation Field Timings in s

MODEL	10k	50k	200k	800k
ARMADILLO	0.3	1.2	6.3	33.9
CUBE	0.11	0.5	2.8	18.5

Table 2. Parametrization Field Timings

MODEL	10k	50k	200k	800k
ARMADILLO	1.3	5.3	21.4	100.3
CUBE	0.15	0.9	6.7	55.1

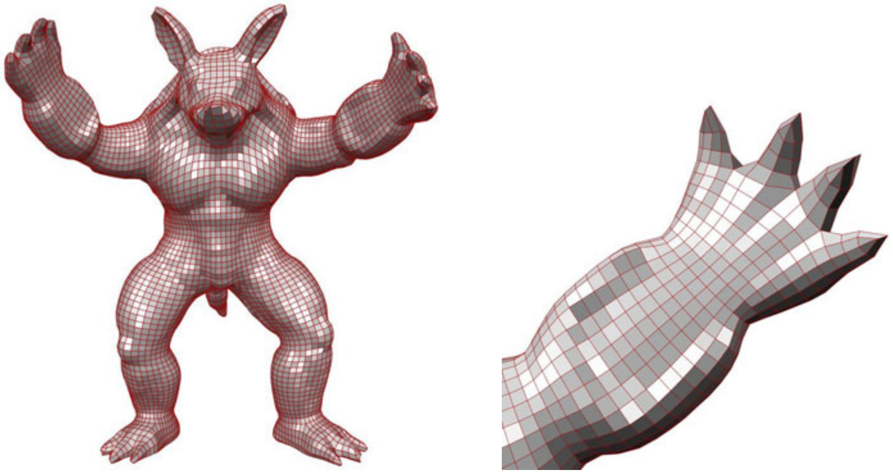


Fig. 3. (left) a quadrangulation of the ARMADILLO designed in an interactive session. (right) close up of the right hand where valence two singularities at the finger tips were created manually.

almost linearly on the number of triangles, enabling very large inputs. A similar behavior can be observed for the parametrization problem in Table 2. The algorithm behaves sensible to the geometric input complexity and nicely adapts to situations of different difficulty which is due to the simultaneous rounding approach.

To underline the importance of the fill-in reducing reordering we did a separate experiment where the PINION model (Figure 2) with many sharp features was parametrized, leading to a huge set of dependent integer constraints. By applying the reordering the computation took $1.3s$ and the system matrix had $418k$ nonzero entries compared to a much slower runtime of $7.4s$ and $581k$ nonzero entries without the reordering.

Besides the performance our algorithm offers a nice flexibility due to the convenient and robust handling of linear constraints as underlined by the following experiment.

Flexibility: Often designers are not satisfied with the result of fully automatic quadrangulation algorithms because they want additional symmetries or structures alleviating animation. Therefore we extended the method of [6] by an interactive manipulation mode where additional (linear) constraints can be iteratively provided by the user. After the fully automatic computation the user is able to (1) change (move, add, remove) singularities, (2) connect singularities by a parametric line to improve the high-level structure of the quadrangulation like used in [15] or (3) add element orientation or alignment constraints. It turned out that such an interaction could be provided easily due to the available robust handling of linear constraints. In the extreme case of specifying all singularities within the orientation field our method is equivalent to [11] and [14]. However in contrast to them our approach is flexible enough to compute solutions for an arbitrary number of known singularities perfectly supporting an interactive design approach. Figure 3 shows the result of an interactive session where the user provided a few orientations. Furthermore some singularities (of the automatic solution) at the hand of the ARMADILLO were merged into valence 2 singularities to capture the spiky shape of the fingers without requiring a very small edge length.

5 Conclusion

In this article we presented the technical details of our mixed-integer approximation algorithm for linearly constrained quadratic mixed-integer problems. By identifying suitable algorithm settings for a given class of optimization problems high efficiency combined with sufficient accuracy is achieved as illustrated by the quadrangulation example. In the future we would like to apply our algorithm to more mixed-integer problems from Geometry Processing and explore further strategies which extend and generalize the idea of simultaneous rounding.

Acknowledgments. This work has been supported by the UMIC Research Centre, RWTH Aachen University.

References

1. Floudas, C.A.: Nonlinear and Mixed Integer Optimization: Fundamentals and Applications. Oxford University Press, New York (1995)
2. Nowak, I.: Relaxation and Decomposition Methods for Mixed Integer Nonlinear Programming. Birkhäuser, Basel (2005)
3. Gupta, O.K., Ravindran, A.: Branch and Bound Experiments in Convex Nonlinear Integer Programming. Manage Sci. 31(12), 1533–1546 (1985)
4. Quesada, I., Grossmann, I.E.: An LP/NLP Based Branch and Bound Algorithm for Convex MINLP Optimization Problems. Computer Chem. Eng. 16, 937–947 (1992)
5. Westerlund, T., Petersson, F.: A Cutting Plane Method for Solving Convex MINLP Problems. Computers Chem. Eng. 19, 131–136 (1995)
6. Bommes, D., Zimmer, H., Kobbelt, L.: Mixed-integer quadrangulation. ACM Trans. Graph. 28(3), 1–10 (2009)
7. Chen, Y., Davis, T.A., Hager, W.W., Rajamanickam, S.: Algorithm 8xx: Cholmod, supernodal sparse cholesky factorization and update/downdate. Technical Report TR-2006-005, University of Florida (2006)

8. Demmel, J.W., Eisenstat, S.C., Gilbert, J.R., Li, X.S., Liu, J.W.H.: A supernodal approach to sparse partial pivoting. SIAM J. Matrix Analysis and Applications 20(3), 720–755 (1999)
9. Kaelberer, F., Nieser, M., Polthier, K.: Quadcover - surface parameterization using branched coverings. Computer Graphics Forum 26(3), 375–384 (2007)
10. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, Heidelberg (1999)
11. Ray, N., Vallet, B., Li, W.C., Lévy, B.: N-symmetry direction field design. ACM Transactions on Graphics (2008); presented at SIGGRAPH
12. Renard, Y.: gmm++, Generic Matrix Methods (2003), `http://home.gna.org/getfem/gmm_intro.html`
13. Ringertz, U.T.: On methods for discrete structural optimization. Engineering Optimization 13(1), 47–64 (1988)
14. Crane, C., Desbrun, M., Schröder, P.: Trivial Connections on Discrete Surfaces. Computer Graphics Forum (SGP) 29(5), 1525–1533 (2010)
15. Myles, A., Pietroni, N., Kovacs, D., Zorin, D.: Feature-aligned T-meshes. ACM Trans. Graph. 29(4), 1–11 (2010)
16. Yu, X., Zhang, S., Johnson, E.: A discrete post-processing method for structural optimization. Engineering with Computers 19(2), 213–220 (2003)
17. Groenwold, A.A., Stander, N., Snyman, J.A.: A pseudo-discrete rounding method for structural optimization. Structural and Multidisciplinary Optimization 11(3), 218–227 (1996)

Non-degenerate Developable Triangular Bézier Patches

Alicia Cantón and Leonardo Fernández-Jambrina

ETSI Navales, Universidad Politécnica de Madrid,
28040-Madrid, Spain
{alicia.canton,leonardo.fernandez}@upm.es
http://dcain.etsin.upm.es

Abstract. In this talk we show a construction for characterising developable surfaces in the form of Bézier triangular patches. It is shown that constructions used for rectangular patches are not useful, since they provide degenerate triangular patches. Explicit constructions of non-degenerate developable triangular patches are provided.

Keywords: Developable surfaces, triangular patches.

1 Introduction

It is well known that developable surfaces play an important role in design in several branches of industry, such as naval and textile. Even architectural structures have been designed using developable surfaces. In these industries surfaces are designed which mimic properties of the materials that are used in production, which are intended to be deformed from plane sheets of metal or cloth just by folding, cutting or rolling, but not stretching. This sort of industrial procedures are less expensive or do not alter the properties of the material and therefore developable surfaces are favoured.

In spite of their importance, developable surfaces are not easy to design within the standard framework of NURBS surfaces. The null gaussian curvature condition is a cubic expression in the parametrization of the surface and can be solved analitically just for low degrees.

This does not mean that NURBS developable surfaces have not been used in design. On the contrary, pieces of plane, cylinders and cones have been used extensively. However, the general case of developable surfaces [1,2], tangent surfaces, has not received the same attention, though it is by large the most important case of developable surfaces.

Since the seminal papers by Mancewicz and Frey [3], Frey and Bindschadler [4] at General Motors, several approaches have been used to cope with developable surfaces:

– Solving null curvature equations for low degrees: papers by Aumann [5], Lang and Röschel, [6], Chalfant and Maekawa [7].

– Projective geometry methods: planes and points are exchanged using duality: Bodduluri and Ravani [8], Pottmann and Farin [9], Pottmann and Wallner [10].
– Based on the de Casteljau algorithm: Chu and Séquin [11], Aumann [12], [13] and Fernández-Jambrina [14].

This last approach has been profitable for obtaining results with tensor product patches of developable surfaces and in this paper we would like to derive an extension to triangular patches.

The paper is organised as follows. Section 2 is devoted to ruled triangular patches. Section 3 provides a quick overview of differential geometry of developable surfaces. Section 4 reviews construction of tensor product developable surfaces. This approach is extended to triangular patches in Section 5. Finally, cylindrical and conical triangular patches are described in Section 6.

2 Ruled Triangular Bézier Patches

Triangular Bézier patches are an alternative to tensor product patches for designing polynomial surfaces. Instead of dealing with parametrizations of degree n_1 in a variable and degree n_2 in the other one, triangles are parametrizations of overall degree n.

Triangular Bézier patches of degree n (cfr. for instance [15] for a review) are surfaces parametrised by

$$b(u,v,w)=\sum_{i+j+k=n}\frac{n!}{i!j!k!}u^iv^jw^kb_{ijk},\quad u+v+w=1,\quad 0\le u,v,w\le 1\ ,$$

for a control net $\{b_{ijk}:\ i+j+k=n,\ 0\le i,j,k\le n\}$ of $(n+2)(n+1)/2$ vertices.

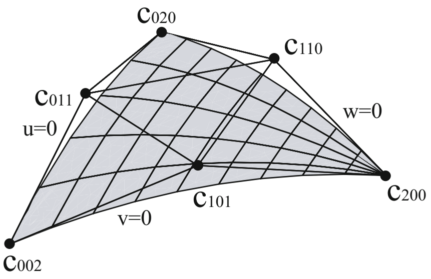


Fig. 1. Bézier triangle of degree two

The surface patch is bounded by three curves of degree n (see Fig. 1) located at $u=0$, $v=0$, $w=0$ and their respective control polygons are given by $\{b_{0jn-j}:\ j=0,\dots,n\}$, $\{b_{i0n-i}:\ i=0,\dots,n\}$, $\{b_{in-i0}:\ i=0,\dots,n\}$.

We are interested in triangular patches of ruled surfaces interpolating linearly between two curves of degree n parametrised by $c(u)$ and $d(v)$, $u, v \in [0,1]$, intersecting at $c(0) = d(0)$, with control polygons $\{c_0, \ldots, c_n\}$ and $\{d_0, \ldots, d_n\}$, so that

$$b(u, 0, 1-u) = c(u), \qquad b(0, v, 1-v) = d(v) \ .$$

The boundary of the patch is formed then by both curves and a straight segment at $w = 0$ linking the ending points of the curves, c_n and d_n. Obviously they have to share the other end, $c_0 = d_0$.

Hence we already know the outer lines of the control net,

$$b_{i0n-i} = c_i,\ i = 0, \ldots, n, \qquad b_{0jn-j} = d_j,\ j = 0, \ldots, n \ ,$$

and by the linear precision property,

$$b_{in-i0} = \frac{i}{n} c_n + \frac{n-i}{n} d_n,\ i = 0, \ldots, n \ ,$$

so that $b(1-v, v, 0) = (1-v)\, c_n + v\, d_n$ traces a straight segment. Hence, we have to prescribe just the inner vertices of the control net.

Since the surface is ruled we require that constant $w = W$ lines on the surface must be straight lines. In order to simplify the analysis, we extend the patch from $u + v + w = 1$ to $u + v = 1$, so that these lines are parametrised as

$$r_W(u) \ = \ b(u, 1-u, W) = \sum_{k=0}^{n} \binom{n}{k} W^k r_k(u) \ ,$$

$$r_k(u) := \sum_{i=0}^{N_k} \binom{N_k}{i} u^i (1-u)^{N_k - i} b_{ijk} \ ,$$

denoting $N_k := n - k$

Since $\{1, W, \cdots, W^n\}$ are linearly independent polynomials, if $r_W(u)$ is to be the affine parametrization of a straight segment for all values of W, every $r_k(u)$ must be the affine parametrization of a straight segment. We consider just the case of general values of the vertices b_{ijk}. It is clear that, as it happens for tensor product patches [16], for special positions of the vertices other solutions could be feasible. But we are interested just in the general case.

Hence, by the linear precision property, for eack k, the vertices

$$\{d_{n-k} = b_{0n-kk}, b_{1n-k-1k}, \cdots, b_{n-k-11k}, b_{n-k0k} = c_{n-k}\}$$

must be equally spaced in order to have linear parametrizations of segments,

$$b_{in-k-ik} = \frac{i}{n-k} c_{n-k} + \frac{n-k-i}{n-k} d_{n-k},\ i = 0, \ldots, n-k \ ,$$

as we checked already for $k = 0$.

Proposition 1. *A Bézier triangular patch of degree n parametrised as $b(u,v,w)$ and bounded by two curves $c(u)$, $d(v)$ of degree n intersecting at $c(0)=d(0)$, with control polygons $\{c_0,\dots,c_n\}$ and $\{d_0,\dots,d_n\}$, so that $b(u,0,1-u)=c(u)$ and $b(0,v,1-v)=d(v)$ is a ruled surface if its control net is given by*

$$b_{ijk}=\frac{ic_{i+j}+jd_{i+j}}{i+j},\ i+j+k=n,\qquad b_{00n}=c_0=d_0\ .$$

That is, the *diagonal* lines of the control net are formed by points which are equally spaced between vertices of the curves with the same index. An example may be seen in Fig. 2.

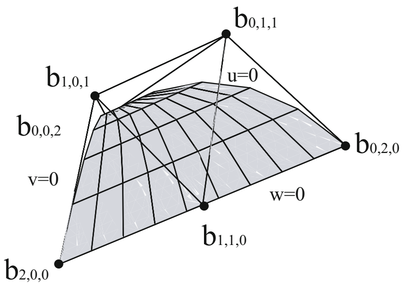


Fig. 2. Ruled Bézier triangle of degree two

For instance, for a triangle of degree four we get a control net

$$\begin{array}{ccccc}
c_0=d_0 & d_1 & d_2 & d_3 & d_4\\
c_1 & \frac{c_2+d_2}{2} & \frac{c_3+2d_3}{3} & \frac{c_4+3d_4}{4} & \\
c_2 & \frac{2c_3+d_3}{3} & \frac{c_4+d_4}{2} & & \\
c_3 & \frac{3c_4+d_4}{4} & & & \\
c_4 & & & &
\end{array}\ .$$

As a counterexample, let us consider a Bézier triangle of degree two, bounded by two curves, which provide every vertex of the control net but b_{110}. If we choose this point aligned with c_2 and d_2, but not in the middle of the segment, it is easy to check that constant w lines are not straight.

Triangular ruled patches may be related to usual explicit ruled parametrizations of surfaces,

$$B(U,V)=(1-V)\,c(U)+V\,d(V)\ ,\quad U,V\in[0,1]\ ,$$

by a change of coordinates,

$$\left.\begin{array}{l}u=U(1-V)\\ v=UV\\ w=1-U\end{array}\right\}\Rightarrow\left\{\begin{array}{l}U=1-w=u+v\\ V=\dfrac{v}{1-w}=\dfrac{v}{u+v},\end{array}\right.$$

which allow us to write down the parametrization of the ruled triangular patch in terms of the parametrizations of the curves,

$$b(u,v,1-u-v) = \frac{u\,c(u+v) + v\,d(u+v)}{u+v} \ .$$

3 Developable Surfaces

Developable surfaces are ruled surfaces with null gaussian curvature [1,2]. Gaussian curvature, K, of a surface parametrised by $b(u,v)$ with unitary normal vector $\nu = b_u \times b_v/\|b_u \times b_v\|$ is defined as the quotient of the determinants of its second, B, and first, G, fundamental forms,

$$G = \begin{pmatrix} b_u \cdot b_u & b_u \cdot b_v \\ b_v \cdot b_u & b_v \cdot b_v \end{pmatrix} , \quad B = \begin{pmatrix} \nu \cdot b_{uu} & \nu \cdot b_{uv} \\ \nu \cdot b_{vu} & \nu \cdot b_{vv} \end{pmatrix} , \quad K(u,v) = \frac{\det B(u,v)}{\det G(u,v)} ,$$

but Gauss' *Theorema Egregium* states that K may be written in terms of the first fundamental form and its derivatives. Since the first fundamental form determines angles, lengths and areas on the surface, gaussian curvature is invariant under transformations, isometries, which preserve such features.

Starting with the usual parametrization of a ruled surface bounded by two curves $c(u)$, $d(u)$,

$$b(u,v) = (1-v)c(u) + vd(u) \ , \quad u,v \in [0,1] \ , \tag{1}$$

since the second derivative $b_{vv}(u,v)$ is null, the determinant of the second fundamental form is negative and hence the gaussian curvature is negative or null at every point of a ruled surface. The determinant of the first fundamental form is positive, since this form is just the inner product of $\mathbb{R}^3$ restricted to tangent vectors to the surface.

Hence, developable surfaces are characterised by vanishing $\nu \cdot b_{uv}$ at every point, that is,

$$\begin{aligned} 0 &= (d'(u) - c'(u)) \cdot ((1-v)c'(u) + vd'(u)) \times (d(u) - c(u)) \\ &= d'(u) \cdot c'(u) \times (d(u) - c(u)) \ . \end{aligned}$$

This provides a useful and geometrical characterization of developable surfaces:

Proposition 2. *A ruled surface parametrised as (1) is developable if and only if the vector* $\mathbf{v}(u) = d(u) - c(u)$*, linking the points* $d(u)$*,* $c(u)$*, and the velocities of the curves at these points are coplanary for every value of* u*.*

Or put in another way, the tangent plane is the same for all points along the straight line (generatrix or ruling of the surface) linking $d(u)$ with $c(u)$.

This means that we may write one of those velocities as a linear combination of the other two vectors,

$$c'(u) = \lambda(u)\mathbf{v}(u) + \mu(u)\mathbf{v}'(u) \ . \tag{2}$$

This is useful for classifying developable surfaces:

1. Planar surfaces: Pieces of planes are the trivial case of surfaces of null curvature.
2. Cylindrical surfaces: Ruled surfaces in which all straight lines (rulings) are parallel. For them $\mathbf{v}(u)$ is parallel to $\mathbf{v}'(u)$.
3. Conical surfaces: Ruled surfaces in which all rulings meet at a point named vertex.
4. Tangent surfaces: Ruled surfaces formed by all tangent lines to a given curve.

The latter is the most general case, since every non-cylindrical surface may be shown to be either a tangent surface to a curve or, fulfilling additional conditions, a conical surface:

Let us perform a change of base curve from $c(u)$ by gliding it along the rulings to $\tilde{c}(u) = c(u) - \mu(u)\mathbf{v}(u)$,

$$\tilde{c}'(u) = c'(u) - \mu'(u)\mathbf{v}(u) - \mu(u)\mathbf{v}'(u) = (\lambda(u) - \mu'(u))\,\mathbf{v}(u) \ .$$

In the general case, the velocity $\tilde{c}'(u)$ is parallel to the rulings of vector $\mathbf{v}(u)$, that is, the surface is a tangent surface to the curve $\tilde{c}(u)$. Only in the restrictive case for which $\lambda(u) = \mu'(u)$, $\tilde{c}'(u) \equiv 0$, the new base curve reduces to a point, the vertex of a cone.

4 Tensor Product Developable Patches

In order to describe Bézier developable surfaces we start by considering a ruled surface interpolated between two polynomial curves of degree n, $c(u)$, $d(u)$, defined by their respective control polygons, $\{c_0, \dots, c_n\}$, $\{d_0, \dots, d_n\}$,

$$c(u) = \sum_{i=0}^{n} c_i B_i^n(u), \qquad d(u) = \sum_{i=0}^{n} d_i B_i^n(u) \ ,$$

in terms of the Bernstein polynomials of degree n, or the de Casteljau algorithm [17],

$$\begin{aligned} c_i^{1)}(u) &= (1-u)c_i(u) + uc_{i+1}(u), \ i = 0, \dots, n-1 \ , \\ c_i^{r)}(u) &= (1-u)c_i^{r-1)}(u) + uc_{i+1}^{r-1)}(u) \ i = 0, \dots, n-r \ , \\ c(u) := c_0^{n)}(u) &= (1-u)c_0^{n-1)}(u) + uc_1^{n-1)}(u) \ . \end{aligned} \tag{3}$$

The derivative of the curves,

$$c'(u) = n\left(c_1^{n-1)}(u) - c_0^{n-1)}(u)\right), \quad d'(u) = n\left(d_1^{n-1)}(u) - d_0^{n-1)}(u)\right) \ ,$$

may be written as a difference between the two last-but-one points in the de Casteljau algorithm.

Hence the vectors $c'(u)$, $d'(u)$, $d(u) - c(u)$ are barycentric combinations of the points $c_0^{n-1)}(u)$, $c_1^{n-1)}(u)$, $d_0^{n-1)}(u)$, $d_1^{n-1)}(u)$. Since we have already seen that the ruled surface is developable if and only if these vectors are coplanary, the developability condition for a Bézier ruled surface may be restated in terms of these:

Proposition 3. *The ruled surface interpolating between two Bézier curves of degree n, defined by their respective control polygons, $\{c_0,\dots,c_n\}$, $\{d_0,\dots,d_n\}$ is developable if and only if the points $c_0^{n-1)}(u)$, $c_1^{n-1)}(u)$, $d_0^{n-1)}(u)$, $d_1^{n-1)}(u)$ are coplanary.*

That is, there exist coefficients $\Lambda(u)$, $M(u)$, such that

$$(1-\Lambda(u))\,c_0^{n-1)}(u)+\Lambda(u)c_1^{n-1)}(u)=(1-M(u))\,d_0^{n-1)}(u)+M(u)d_1^{n-1)}(u)\ . \tag{4}$$

This way of writing the linear combination excludes the conical case. However, it does not hinder our goal of coping with the generic case.

We may gain insight into this result by rewriting it in terms of blossoms,

$$\begin{aligned} c_i^{1)}[u_1] &:= c_i^{1)}(u_1)=(1-u_1)c_i+u_1c_{i+1}, \quad i=0,\dots,n-1\ ,\\ c_i^{r)}[u_1,\dots,u_r] &:= (1-u_r)c_i^{r-1)}[u_1,\dots,u_{r-1}]+u_rc_{i+1}^{r-1)}[u_1,\dots,u_{r-1}]\ ,\\ c[u_1,\dots,u_n] &:= c_0^{n)}[u_1,\dots,u_n]\ , \quad i=0,\dots,n-r, \quad r=1,\dots,n\ , \end{aligned} \tag{5}$$

since the linear combinations of the points,

$$c_0^{n-1)}(u)=c[u^{<n-1>},0]\ , \qquad c_1^{n-1)}(u)=c[u^{<n-1>},1]\ ,$$

can be written in a rather compact form, taking into account that blossoms are multi-affine,

$$c[u^{<n-1>},\Lambda(u)]=d[u^{<n-1>},M(u)]\ . \tag{6}$$

We have therefore characterised developability of a rational ruled surface in terms of blossoms:

Theorem 1. *Two Bézier curves $c(u)$, $d(u)$ with control polygons $\{c_0,\dots,c_n\}$, $\{d_0,\dots,d_n\}$ define a generic developable surface if and only if their respective blossoms are related by*

$$c[u^{<n-1>},\Lambda(u)]=d[u^{<n-1>},M(u)]$$

The simplest case which can be analysed is the one of constant coefficients Λ, M,

$$c[u^{<n-1>},\Lambda]=d[u^{<n-1>},M]\ ,$$

which is the family of developable surfaces found by Aumann [12], though in that paper the key issue was the use of an affine transformation between adjacent cells of the control net of the surface.

This expression states the equality of two $(n-1)$-atic forms, which is equivalent to the equality of the respective symmetric $(n-1)$-affine forms, since the correspondence between blossoms and parametrizations is one-to-one,

$$c[u_1,\dots,u_{n-1},\Lambda]=d[u_1,\dots,u_{n-1},M]\ \ . \tag{7}$$

We may draw information about the control net applying it to sequences of zeros and ones, taking into account that the vertices are recovered as

$$c_j = c[0^{<n-j>}, 1^{<j>}] \ ,$$

$$(1-\Lambda)c_j + \Lambda c_{j+1} = (1-M)d_j + M d_{j+1} \ , \quad j = 0, \ldots, n-1 \ ,$$

stating that the cells of the control net of the surface are planar and share the same linear combination between vertices.

These conditions may be solved recursively,

$$d_n = \left(\frac{M-1}{M}\right)^n d_0 + \frac{1-\Lambda}{M}\left(\frac{M-1}{M}\right)^{n-1} c_0 + \frac{M-\Lambda}{M^2}\sum_{i=1}^{n-1}\left(\frac{M-1}{M}\right)^{n-i-1} c_i + \frac{\Lambda}{M} c_n \ ,$$

in order to relate the first and last rulings of the patch with the vertices of the control polygon of the curve $c(u)$,

$$\begin{aligned} d_n - c_n = \frac{M-\Lambda}{M}\left(\left(\frac{M-1}{M}\right)^{n-1} c_0 + \frac{1}{M}\sum_{i=1}^{n-1}\left(\frac{M-1}{M}\right)^{n-i-1} c_i - c_n\right) \\ + \left(\frac{M-1}{M}\right)^n (d_0 - c_0) \ , \end{aligned} \tag{8}$$

or even its sides,

$$d_n - c_n = \left(\frac{M-1}{M}\right)^n (d_0 - c_0) + \frac{\Lambda - M}{M}\sum_{i=0}^{n-1}\left(\frac{M-1}{M}\right)^{n-i-1} \Delta c_i \ , \tag{9}$$

denoting $\Delta c_i = c_{i+1} - c_i$.

This construction of developable Bézier surfaces can be used to solve an interpolation problem [12]:

"Given a Bézier curve $c(u)$ of degree n and two straight lines l_0 and l_1 passing through the endpoints of $c(u)$, find a developable surface $b(u,v)$ through $c(u)$ ($b(u,0) = c(u)$) with l_0 and l_1 as first and last ruling ($l_0 : \ c(0,v)$, $l_1 : \ c(1,v)$)."

Depending on the position of the rulings l_0, l_1 we have three possible solutions to this problem:

– If l_0, l_1 are parallel, we may construct a cylinder through the curve $c(u)$ with rulings parallel to l_0 and l_1.
– If l_0, l_1 meet at one point V, we may construct a cone through $c(u)$ and vertex at V.
– If l_0, l_1 are neither parallel nor meeting at one point, we may resort to Aumann's construction ([8]),

$$\begin{aligned} d_n - c_n &= \frac{\Lambda - M}{M}\big(c_n - a(M)\big) + \left(\frac{M-1}{M}\right)^n (d_0 - c_0) \ , \\ a(M) &:= \left(\frac{M-1}{M}\right)^{n-1} c_0 + \frac{1}{M}\sum_{i=1}^{n-1}\left(\frac{M-1}{M}\right)^{n-i-1} c_i \ , \end{aligned} \tag{10}$$

relating a vector on l_0, $d_0 - c_0 = \sigma \mathbf{v}$, and a vector on l_1, $d_n - c_n = \tau \mathbf{w}$ with a vector which is a barycentric combination of the vertices of the control polygon of the curve $c(u)$, $c_n - a(M)$.

This imposes a restriction on the value of M through an equation of degree $n-1$,

$$\det(d_0 - c_0, d_n - c_n, a(M) - c_n) = 0 \ .$$

If M_0 is a solution of this equation, we may reckon the coefficients of the linear combination,

$$a(M_0) = c_n + \alpha_0 \mathbf{v} + \beta_0 \mathbf{w} \ ,$$

solving the linear system using Cramer's rule,

$$\alpha_0 = \frac{\det(a(M_0) - c_n, \mathbf{w}, \mathbf{N})}{\det(\mathbf{v}, \mathbf{w}, \mathbf{N})} \ , \qquad \beta_0 = \frac{\det(\mathbf{v}, a(M_0) - c_n, \mathbf{N})}{\det(\mathbf{v}, \mathbf{w}, \mathbf{N})} \ ,$$

where $\mathbf{N} = \mathbf{v} \times \mathbf{w}$ is a vector that completes a linear basis $\{\mathbf{v}, \mathbf{w}, \mathbf{N}\}$.

Hence, equation (10) is written as

$$\tau_0 \mathbf{w} = \frac{M_0 - \Lambda}{M_0} (\alpha_0 \mathbf{v} + \beta_0 \mathbf{w}) + \left(\frac{M_0 - 1}{M_0} \right)^n \sigma_0 \mathbf{v} \ ,$$

from which we may read the coefficients σ_0 and τ_0 that determine the ends of the rulings,

$$\sigma_0 = \frac{\Lambda - M_0}{M_0 - 1} \left(\frac{M_0}{M_0 - 1} \right)^{n-1} \alpha_0 \ , \qquad \tau_0 = \frac{M_0 - \Lambda}{M_0} \beta_0 \ . \tag{11}$$

The coefficient Λ remains a free parameter and may be fixed by choosing either d_0 along l_0 or c_n along l_1, but not both. This problem may be avoided by elevating the degree of the surface, stretching the surface patch along the rulings $d(u) - c(u)$ [13].

If we have already made use of Λ for fixing d_n, this may be accomplished by multiplying this vector by a linear factor $(1 - A)u + A$, so that the new surface patch

$$\tilde{b}(u, v) = c(u) + v \left(d(u) - c(u) \right) \big((1 - A)u + A \big) \ ,$$

is bounded by the curves $c(u)$ and $\tilde{d}(u) = c(u) + \big((1 - A)u + A \big) \left(d(u) - c(u) \right)$ and we may use the coefficient A for choosing the end of the other ruling,

$$\tilde{d}_0 = \tilde{d}(0) = c_0 + A \left(d_0 - c_0 \right) .$$

As we see in the next section, this construction is useful for designing developable triangular patches.

5 Triangular Developable Patches

We may try to use Aumann's family of developable surfaces to construct triangular developable surfaces limited by two curves of degree n and control polygons $\{c_0, \ldots, c_n\}$, $\{d_0, \ldots, d_n\}$. The first cell of the control net is restricted by

$$(1-\Lambda)c_0+\Lambda c_1=(1-M)d_0+Md_1\ ,$$

but since the curves intersect at $c_0=d_0$, the three points must be aligned,

$$d_1=\left(1-\frac{\Lambda}{M}\right)c_0+\frac{\Lambda}{M}c_1\ .$$

This is a severe restriction, since it implies that the initial velocities of the curves must be parallel with this construction. An example may be seen in Fig. 3.

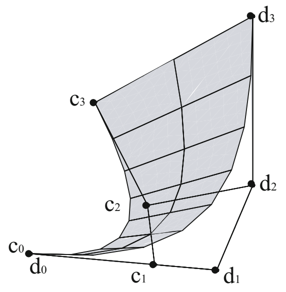


Fig. 3. Degenerate triangular developable patch

Therefore, Aumann's family of developable surfaces does not seem to be a good starting point for designing triangular patches. However, we may use them as an auxiliary patch for constructing them.

Though we do not know the direction of the ruling at the initial vertex of the triangular patch, we may use Aumann's construction to design a tensor product developable patch through a curve $c(u)$ of degree n and control polygon $\{c_0,\dots,c_n\}$ and fixing the last ruling by the choice of d_n,

$$b(u,v)=c(u)+v\,\mathbf{v}(u)\ ,\quad \mathbf{v}(u)=d(u)-c(u)\ .$$

We fix the unknown vertex d_0 by shortening the patch along the rulings of direction $\mathbf{v}(u)$,

$$\tilde{b}(u,v)=c(u)+v\,u\mathbf{v}(u)\ ,$$

so that the new bounding curve $\tilde{d}(u)=\tilde{b}(u,1)$ meets $c(u)$ at c_0.

The velocity of the $v=$ *const.* curves is given by

$$\frac{\partial\tilde{b}(u,v)}{\partial u}=c'(u)+v\,\mathbf{v}(u)+v\,u\mathbf{v}'(u)\ .$$

In particular, at the beginning of the curve $\tilde{d}(u)$,

$$\tilde{d}'(0)=c'(0)+\mathbf{v}(0)=n(c_1-c_0)+(d_0-c_0)\ ,$$

we learn that we may fix the auxiliary initial ruling by prescribing the initial velocity of the bounding curve $\tilde{d}(u)$,

$$d_0 = (n+1)c_0 - nc_1 + \tilde{d}'(0) \ .$$

Hence, by this procedure it is possible to find triangular developable patches with boundary on $c(u)$ and the ruling $\overline{d_n c_n}$ fixing the value of $d'(0)$ and making use of Aumann's construction.

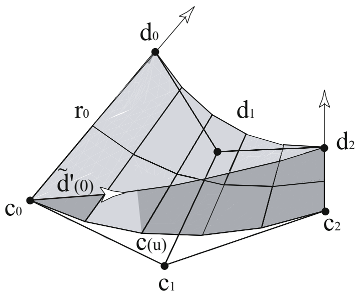


Fig. 4. Stretching a tensor product patch to a triangular patch

6 Cylindrical and Conical Triangular Patches

Triangular patches of cylinders and cones are easier to construct than tangent surfaces.

Cylinders bounded by a curve $c(u)$ of degree n and rulings parallel to a constant vector $\mathbf{v}$ are parametrised as

$$b(u,v) = c(u) + vf(u)\,\mathbf{v} \ ,$$

where $f(u)$ is a polynomial vanishing at $u = 0$. The other bounding curve is $d(u) = b(u,1)$. An example is shown in Fig. 5.

Hence, the only requirement for building a cylindrical triangular patch is that the vertices of the control polygons of the bounding curves, $\{c_0,\dots,c_n\}$, $\{d_0,\dots,d_n\}$ must lie on parallel lines,

$$\overrightarrow{c_1 d_1} \parallel \cdots \parallel \overrightarrow{c_n d_n} \ ,$$

except for the first pair which coalesce to a single point, $c_0 = d_0$.

Cones through a curve $c(u)$ and with vertex on a point a may be parametrised as

$$b(u,v) = c(u) + v\,\mathbf{v}(u) \ , \quad \mathbf{v}(u) = c(u) - a \ .$$

Hence, if $c(u)$ is a curve of degree n, a curve $d(u)$ at $v = \mathit{const.}$ is also of the same degree. Since such curves are scaled copies of $c(u)$, their control polygons must have sides proportional to the ones of the original curve,

$$\overrightarrow{d_i d_{i-1}} = \alpha \overrightarrow{c_i c_{i-1}} \ , \quad i = 1,\dots,n \ ,$$

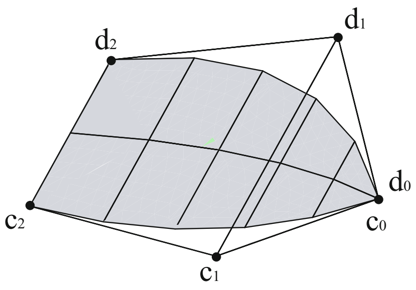


Fig. 5. Cylindrical triangular patch

being $\{d_0, \ldots, d_n\}$ the control polygon of the second curve.

We may proceed as we did for tangent surfaces in order to get triangular conical patches of degree $n + 1$. We shorten the patch linearly along the ruling so that the first generatrix is reduced to a single point,

$$\tilde{b}(u, v) = c(u) + vu\,\mathbf{v}(u) \ .$$

The degree of the bounding curve $\tilde{d}(u) = \tilde{b}(u, 1)$ is $n+1$. An example is shown in Fig. 6.

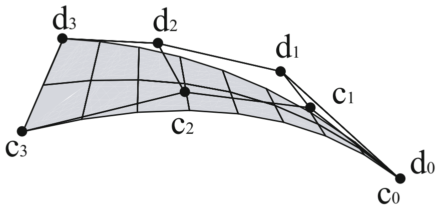


Fig. 6. Conical triangular patch

7 Conclusions

In this paper control nets for ruled triangular Bézier patches bounded by two curves and a straight line have been constructed. It has been shown that Aumann's construction, which has been useful for designing general developable surfaces with tensor product patches, renders degenerate triangular patches. A construction grounded on degree elevation has been devised for bypassing this problem and producing nondegenerate triangular Bézier developable surfaces. This construction has been used for providing solutions to the problem of interpolating a triangular developable surface based on a curve and the last ruling of the surface, knowing the initial velocity of the other bounding curve.

Acknowledgments. The present work has been supported by Spanish Ministry of Science grant MTM2011-22886.

References

1. Postnikov, M.M.: Lectures in Geometry: Linear Algebra and Differential Geometry. "Nauka", Moscow (1979)
2. Struik, D.J.: Lectures on classical differential geometry, 2nd edn. Dover Publications Inc., New York (1988)
3. Mancewicz, M., Frey, W.: Developable surfaces: properties, representations and methods of design. Technical report, GM Research Publication GMR-7637 (1992)
4. Frey, W., Bindschadler, D.: Computer aided design of a class of developable bézier surfaces. Technical report, GM Research Publication R&D-8057 (1993)
5. Aumann, G.: Interpolation with developable Bézier patches. Comput. Aided Geom. Design 8(5), 409–420 (1991)
6. Lang, J., Röschel, O.: Developable $(1, n)$-Bézier surfaces. Comput. Aided Geom. Design 9(4), 291–298 (1992)
7. Chalfant, J., Maekawa, T.: Design for manufacturing using b-spline developable surfaces. J. Ship Research 42(3), 207–215 (1998)
8. Bodduluri, R., Ravani, B.: Design of developable surfaces using duality between plane and point geometries. Computer Aided Design 25(10), 621–632 (1993)
9. Pottmann, H., Farin, G.: Developable rational Bézier and B-spline surfaces. Comput. Aided Geom. Design 12(5), 513–531 (1995)
10. Pottmann, H., Wallner, J.: Approximation algorithms for developable surfaces. Comput. Aided Geom. Design 16(6), 539–556 (1999)
11. Chu, C.H., Séquin, C.H.: Developable bézier patches: properties and design. Computer Aided Design 34(7), 511–527 (2002)
12. Aumann, G.: A simple algorithm for designing developable Bézier surfaces. Comput. Aided Geom. Design 20(8-9), 601–619 (2003); In memory of Professor J. Hoschek
13. Aumann, G.: Degree elevation and developable Bézier surfaces. Comput. Aided Geom. Design 21(7), 661–670 (2004)
14. Fernández-Jambrina, L.: B-spline control nets for developable surfaces. Comput. Aided Geom. Design 24(4), 189–199 (2007)
15. Farin, G.: Triangular bernstein-bézier patches. Comput. Aided Geom. Design 3(2), 83–127 (1986)
16. Juhász, I., Róth, Á.: Bézier surfaces with linear isoparametric lines. Comput. Aided Geom. Design 25(6), 385–396 (2008)
17. Farin, G.: Curves and surfaces for CAGD: a practical guide. Morgan Kaufmann Publishers Inc., San Francisco (2002)

Stable Splitting of Bivariate Splines Spaces by Bernstein-Bézier Methods

Oleg Davydov and Abid Saeed

Department of Mathematics and Statistics,
University of Strathclyde, Glasgow, United Kingdom
{oleg.davydov,abid.saeed}@strath.ac.uk
http://www.mathstat.strath.ac.uk/

Abstract. We develop stable splitting of the minimal determining sets for the spaces of bivariate C^1 splines on triangulations, including a modified Argyris space, Clough-Tocher, Powell-Sabin and quadrilateral macro-element spaces. This leads to the stable splitting of the corresponding bases as required in Böhmer's method for solving fully nonlinear elliptic PDEs on polygonal domains.

Keywords: Fully nonlinear PDE, Monge-Ampère equation, multivariate splines, Bernstein-Bézier techniques.

1 Introduction

Numerical solution of fully nonlinear elliptic partial differential equations is a topic of intensive research and great practical interest, see [2,4]. Since no weak form formulation is available for the equations of this type in general, the standard Galerkin finite element method cannot be applied directly.

Recently, Böhmer [1,2] introduced a general approach that solves the Dirichlet problem for fully nonlinear elliptic equations numerically with the help of a sequence of linear elliptic equations used within an appropriate Newton scheme. These linear elliptic equations can be solved by the finite element method, but the discretisation has to be done by appropriate spaces of C^1 finite elements (splines) that admit a stable splitting into a subspace satisfying zero boundary conditions, and its complement. Such a stable splitting has been developed in [6] for a modified space of the Argyris finite element.

In this paper we systematically study the problem of stable splitting for the spaces of bivariate C^1 splines on triangulations of low degree using the Bernstein-Bézier methods. It turns out that stable splitting can be easily formulated as splitting of the minimal determining sets (MDS). We revisit the modified Argyris space studied in [6] by a different technique, and show that its modification is necessary at least if the convenient MDS splitting approach is used. We also show that Clough-Tocher, Powell-Sabin and quadrilateral macro-element spaces admit the stable splitting and therefore can also be used in the Böhmer's numerical method.

The paper is organised as follows. Section 2 is devoted to an outline of Böhmer's method, whereas Section 3 introduces necessary definitions from the theory of Bernstein-Bézier methods [8], and defines the stable splitting of an MDS. In Section 4 we discuss the stable splitting for the Argyris space and its modification, and Section 5 is devoted to the C^1 macro-element spaces.

2 Böhmer's Method for Fully Nonlinear Elliptic PDEs

2.1 Fully Nonlinear Elliptic Operators

Let Ω be a bounded domain in $\mathbb{R}^n$ and let $G : H^\gamma(\Omega) \to L^2(\Omega)$, $\gamma \geq 2$, be a second order differential operator of the form

$$G(u) = \widetilde{G}(\cdot, u, \nabla u, \nabla^2 u),$$

where $\widetilde{G}$ is a real valued function defined on a domain $\widetilde{\Omega} \times \Gamma$ such that

$$\overline{\Omega} \subset \widetilde{\Omega} \subset \mathbb{R}^n \quad \text{and} \quad \Gamma \subset \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^{n\times n},$$

and $\nabla u, \nabla^2 u$ denote the gradient and the Hessian of u, respectively. The points in $\widetilde{\Omega} \times \Gamma$ are denoted by $w = (x, z, p, r)$, with $x \in \widetilde{\Omega}$, $z \in \mathbb{R}$, $p = [p_i]_{i=1}^n \in \mathbb{R}^n$, $r = [r_{ij}]_{i,j=1}^n \in \mathbb{R}^{n\times n}$, to indicate the product structure of this set.

The operator G is said to be *elliptic* in a subset $\widetilde{\Gamma} \subset \widetilde{\Omega} \times \Gamma$ if the matrix $[\frac{\partial \widetilde{G}}{\partial r_{ij}}(w)]_{i,j=1}^n$ is well defined and positive definite for all $w \in \widetilde{\Gamma}$ [2,7]. If $\widetilde{G}$ is a linear function of (z, p, r) for each fixed x, then G is a linear differential operator. Under suitable restrictions on $\widetilde{G}$, classes of quasilinear and semilinear differential operators are obtained [2, p. 80], but in general G may be fully nonlinear.

In the neighborhood of a fixed function $\hat{u} \in H^\gamma(\Omega)$ the linear elliptic operator $G'(\hat{u})$ is defined by

$$G'(\hat{u})u = \frac{\partial \widetilde{G}}{\partial z}(\hat{w})u + \sum_{i=1}^{n} \frac{\partial \widetilde{G}}{\partial p_i}(\hat{w})\partial^i u + \sum_{i,j=1}^{n} \frac{\partial \widetilde{G}}{\partial r_{ij}}(\hat{w})\partial^i \partial^j u,$$

where $\hat{w} = (x, \hat{u}(x), \nabla \hat{u}(x), \nabla^2 \hat{u}(x))$ is a function of $x \in \Omega$, and ∂^i denotes the partial derivative with respect to the i-th variable. If $G : H^\gamma(\Omega) \to L^2(\Omega)$ is Fréchet differentiable at $\hat{u}$, then $G'(\hat{u}) : H^\gamma(\Omega) \to L^2(\Omega)$ is its Fréchet derivative. If $G'(\hat{u})$ depends continuously on $\hat{u}$ with respect to the linear operator norm, then G is said to be *continuously differentiable* at $\hat{u}$.

Many nonlinear elliptic operators and corresponding equations $G(u) = 0$ are important for applications, for example the *Monge-Ampère equation* for $\Omega \subset \mathbb{R}^2$, given by

$$G_{\mathrm{MA}}(u) := \det(\nabla^2 u) - f(x) = 0, \quad f(x) > 0 \text{ for } x \in \Omega.$$

The operator G_{MA} is fully nonlinear and $G_{\mathrm{MA}}(u) \in L^2(\Omega)$ if u belongs to the Sobolev space $H^{5/2}(\Omega)$ and $f \in L^2(\Omega)$. Moreover, $G_{\mathrm{MA}} : H^\gamma(\Omega) \to L^2(\Omega)$ is continuously differentiable if $\gamma \geq 5/2$.

We consider the Dirichlet problem for the operator G: Find u such that

$$G(u) = 0, \quad x \in \Omega, \tag{1}$$

$$u = \phi, \quad x \in \partial\Omega, \tag{2}$$

where ϕ is a continuous function defined on $\partial\Omega$. Under certain assumptions (including the exterior sphere condition for $\partial\Omega$ and sufficient smoothness of $\widetilde{G}$, satisfied in particular in the above mentioned examples if $f \in C^2(\Omega)$), this problem has a unique solution $u \in C^2(\Omega) \cap C(\overline{\Omega})$ [7, Theorem 17.17]. Note that the Monge-Ampère operator G_{MA} is elliptic in subsets $\widetilde{\Gamma}$ satisfying

$$\widetilde{\Gamma} \subset \widetilde{\Omega} \times \mathbb{R} \times \mathbb{R}^n \times \{r \in \mathbb{R}^{n\times n} : r \text{ is positive definite}\}.$$

Therefore there exists a unique convex solution of $G_{\mathrm{MA}}(u) = 0$, whereas it is known that the Monge-Ampère equation has another, concave solution [3, Chapter 4].

2.2 Spline Spaces and Stable Splitting

As usual in the finite element method, the discretisation of the Dirichlet problem is done with the help of spaces of piecewise polynomial functions (splines). Let $\triangle$ be a triangulation of a polyhedral domain $\Omega \subset \mathbb{R}^n$, that is a partition of Ω into simplices such that the intersection of every pair of simplices is either empty or a common face. The space of multivariate splines of degree d and smoothness r is defined by

$$S_d^r(\triangle) = \{s \in C^r(\Omega) : s|_T \in P_d \text{ for all simplices } T \text{ in } \triangle\}, \tag{3}$$

where $d > r \geq 0$ and P_d is the space of polynomials of total degree d in n variables. Recall that the *star* of a vertex v of $\triangle$, denoted by $\mathrm{star}(v) = \mathrm{star}^1(v)$, is the union of all triangles $T \in \triangle$ attached to v. We define $\mathrm{star}^j(v)$, $j \geq 2$, inductively as the union of the stars of all vertices of $\triangle$ contained in $\mathrm{star}^{j-1}(v)$.

Let $\{\triangle^h\}_{h\in H}$ be a family of triangulations of Ω, where h is the maximum edge length in $\triangle^h$. The triangulations in the family are said to be *quasi-uniform* if there is an absolute constant $c > 0$ such that $\rho_T \geq ch$ for all $T \in \triangle^h$, where ρ_T denotes the radius of the inscribed sphere of the simplex T.

Let $S^h \subset S_d^r(\triangle^h)$ be a linear subspace with basis $s_1, \ldots, s_N$ and dual functionals $\lambda_1, \ldots, \lambda_N$ such that $\lambda_i s_j = \delta_{ij}$. This basis is *stable* and *local* if there are three constants $m \in \mathbb{N}$ and $C_1, C_2 > 0$ independent of h such that (a) $\operatorname{supp} s_k$ is contained in $\mathrm{star}^m(v)$ for some vertex v of $\triangle^h$, (b) $\|s_k\|_{L^\infty(\Omega)} \leq C_1$, $k = 1, \ldots, N$, and (c) $|\lambda_k s| \leq C_2 \|s\|_{L^\infty(\operatorname{supp} s_k)}$, $k = 1, \ldots, N$, for all $s \in S^h$, see [5,6] and [2, Section 4.2.6].

To handle the Dirichlet boundary conditions, the following subspace of S^h is important:

$$S_0^h := \left\{s \in S^h : s|_{\partial\Omega} = 0\right\}.$$

Moreover, the method of solving (1)–(2) proposed in [1,2] requires a *stable splitting* of S^h into a direct sum

$$S^h = S_0^h + S_b^h,$$

such that a stable local basis $\{s_1, \dots, s_N\}$ for S^h can be split into two parts

$$\{s_1, \dots, s_N\} = \{s_1, \dots, s_{N_0}\} \cup \{s_{N_0+1}, \dots, s_N\},$$

where $\{s_1, \dots, s_{N_0}\}$ and $\{s_{N_0+1}, \dots, s_N\}$ are bases for S_0^h and S_b^h, respectively. Note that the space S_b^h is not uniquely defined by the pair S^h, S_0^h. It was shown in [6] (see also [2, Section 4.2.6]) how the stable splitting can be achieved for a modified space of Argyris finite element.

2.3 Böhmer's Method

Let $u = \hat{u}$ be the solution of (1)–(2). According to [1], its approximation $\hat{u}^h \approx \hat{u}$ is sought as a solution of the following problem: Find $\hat{u}^h \in S^h$ such that

$$(G(\hat{u}^h), v^h)_{L^2(\Omega)} = 0 \quad \forall v^h \in S_0^h, \quad \text{and} \tag{4}$$

$$(\hat{u}^h, v_b^h)_{L^2(\partial\Omega)} = (\phi, v_b^h)_{L^2(\partial\Omega)} \quad \forall v_b^h \in S_b^h, \tag{5}$$

where $(\cdot,\cdot)$ denotes the inner products in the respective Hilbert spaces. Since S_0^h and S_b^h are finite dimensional linear spaces, the problem (4)–(5) is equivalent to a system of nonlinear equations with respect to the coefficients of $\hat{u}^h$ in a basis of S^h.

Theorem 1 ([1, Theorem 8.7] and [2, Theorem 5.2]). *Let Ω be a bounded convex polyhedral domain, and let $G : D(G) \to L^2(\Omega)$, with $D(G) \subset H^2(\Omega)$, satisfy Condition H of [2, Section 5.2.3]. Assume that G is continuously differentiable in the neighbourhood of an isolated solution $\hat{u}$ of (1)–(2), such that $\hat{u} \in H^\ell(\Omega)$, $\ell > 2$, and $G'(\hat{u}) : D(G) \cap H_0^1(\Omega) \to L^2(\Omega)$ is boundedly invertible. Furthermore, assume that the spline spaces $S^h \subset S_d^1(\triangle^h)$, $d \geq \ell - 1$, on quasi-uniform triangulations $\triangle^h$ possess stable local bases and stable splitting $S^h = S_0^h + S_b^h$, and include polynomials of degree $\ell - 1$. Then the problem (4)–(5) has a unique solution $\hat{u}^h \in S^h$ as soon as the maximum edge length h is sufficiently small. Moreover,*

$$\|\hat{u} - \hat{u}^h\|_{H^2(\Omega)} \leq Ch^{\ell-2}\|\hat{u}\|_{H^\ell(\Omega)}.$$

In particular, Condition H is satisfied by the Monge-Ampère operators on bounded convex polygonal domains in $\mathbb{R}^2$.

The nonlinear problem (4)–(5) can be solved iteratively by a Newton method [1], where the initial guess $u_0^h \in S^h$ satisfies the boundary condition

$$(u_0^h, v_b^h)_{L^2(\partial\Omega)} = (\phi, v_b^h)_{L^2(\partial\Omega)} \quad \forall v_b^h \in S_b^h,$$

and the sequence of approximations $\{u_k^h\}_{k\in\mathbb{N}}$ of $\hat{u}^h$ is generated by

$$u_{k+1}^h = u_k^h - w^h, \quad k = 0, 1, \dots,$$

with $w^h \in S_0^h$ being the solution of the linear elliptic problem:

$$\text{Find } w^h \in S_0^h \text{ such that } (G'(u_k^h)w^h, v^h)_{L^2(\Omega)} = (G(u_k^h), v^h)_{L^2(\Omega)} \; \forall v^h \in S_0^h.$$

Clearly, w^h can be found by using the standard finite element method. Under some additional assumptions on G, it is proved in [1, Theorem 9.1] that u_i^h converges to $\hat{u}$ quadratically. Note that in the case when $G(u)$ is only conditionally elliptic (e.g. elliptic only for a convex u for Monge-Ampère equation) the ellipticity of the above linear problem is only guaranteed for u_k^h sufficiently close to the exact solution $\hat{u}$.

3 Bernstein-Bézier Techniques

Certain spaces of bivariate C^1 splines with stable local bases and stable splitting required in Böhmer's method have been investigated by nodal techniques in [6]. However, Bernstein-Bézier methods are often preferable. Let us recall some related key concepts here, see [8] for more details.

From now on we only consider the bivariate case. In particular, Ω is a polygonal domain in $\mathbb{R}^2$ and $\triangle$ is a triangulation of Ω.

Given $d \geq 1$, let $D_{d,\triangle} := \bigcup_{T\in\triangle} D_{d,T}$ be the set of *domain points*, where

$$D_{d,T} := \left\{ \xi_{ijk} = \frac{iv_1 + jv_2 + kv_3}{d} \right\}_{i+j+k=d}$$

for each triangle $T := \langle v_1, v_2, v_3 \rangle$ in $\triangle$. Also note that every $v \in \mathbb{R}^2$ can be uniquely represented in the form

$$v = \sum_{i=1}^{3} b_i v_i, \quad \sum_{i=1}^{3} b_i = 1.$$

The triplet (b_1, b_2, b_3) is called the *barycentric coordinates* of v relative to the triangle $T := \langle v_1, v_2, v_3 \rangle$, and

$$B_{ijk}^d(v) := \frac{d!}{i!j!k!} b_1^i b_2^j b_3^k, \quad i + j + k = d,$$

are the *Bernstein-Bézier basis polynomials* of degree d associated with triangle T. Every polynomial p of total degree d can be written uniquely as

$$p = \sum_{i+j+k=d} c_{ijk} B_{ijk}^d,$$

where c_{ijk} are the *Bézier coefficients* of p. For each $s \in S_d^r(\triangle)$ and $\xi = \xi_{ijk} \in D_{d,\triangle}$ we denote by c_ξ the coefficient c_{ijk} of the restriction of s to any triangle

$T \in \triangle$ containing ξ. (Because of the continuity of s the coefficient c_ξ does not depend on the particular choice of such triangle.)

We now introduce two additional notations. We refer to the set

$$R_n(v_1) := \{\xi_{ijk} \in D_{d,\triangle} : i = d - n\}, \quad 0 \le n \le d,$$

of domain points as the *ring* of radius n arround the vertex v_1 and refer to the set

$$D_n(v_1) := \bigcup_{m=0}^{n} R_m(v_1)$$

as the *disk* of radius n arround the vertex v_1.

A key concept for dealing with spline spaces is that of a minimal determining set. Recall that the set $M \subset D_{d,\triangle}$ is a *determining set* for a linear space $S \subset S_d^r(\triangle)$ if

$$s \in S \text{ and } c_\xi = 0 \quad \forall \xi \in M \quad \Rightarrow \quad s = 0,$$

and M is a *minimal determining set* (MDS) for the space S if there is no smaller determining set. Then $\dim S$ equals the cardinality $\#\{M\}$ of M. Let

$$\Gamma_\eta := \{\xi \in M \ : \ c_\eta \text{ depends on } c_\xi\},$$

where we say that c_η depends on c_ξ, $\xi \in M$, if the value of c_η is changed when we change the value of c_ξ. A minimal determining set M for a space S is said to be *local* if there exists an absolute integer constant ℓ not depending on $\triangle$ such that

$$\Gamma_\eta \subset \operatorname{star}^\ell(T_\eta) \quad \forall \eta \in D_{d,\triangle} \backslash M,$$

where T_η is a triangle containing η. And M is called *stable* if there exists a constant K which may depend only on d, ℓ and the smallest angle $\theta_\triangle$ in the triangulation $\triangle$ such that

$$|c_\eta| \le K \max_{\xi \in \Gamma_\eta} |c_\xi| \quad \forall \eta \in D_{d,\triangle} \backslash M.$$

Given a stable local minimal determining set M for $S \subset S_d^r(\triangle)$, a stable local basis $\{s_\xi\}_{\xi \in M}$ for S can be defined by requiring that the Bézier coefficients c_η, $\eta \in M$, of s_ξ satisfy $c_\xi = 1$ and $c_\eta = 0$ for all $\eta \in M \setminus \{\xi\}$, see [8, Section 5.8]. Clearly, a stable splitting of this basis is achieved by an appropriate splitting of the MDS, which leads to the following definition.

Definition 1. *Assume that the space $S \subset S_d^r(\triangle)$ has a stable local MDS M and let*

$$S_0 := \{s \in S \ : \ s|_{\partial\Omega} = 0\}. \tag{6}$$

The MDS M is said to admit a stable splitting *if M is the disjoint union of two subsets $M_0, M_b \subset M$ such that*

$$S_0 = \{s \in S : c_\xi = 0 \ \forall \xi \in M_b\} \tag{7}$$

and M_0 and M_b are stable local MDS for the spaces S_0 and S_b, respectively, where

$$S_b := \{s \in S : c_\xi = 0 \ \forall \xi \in M_0\}. \tag{8}$$

Note that if M is a stable local MDS, and $M = M_0 \cup M_b$ is a disjoint union, then it is a stable splitting as soon as (7) holds. Indeed, assume (7) is correct. If $s \in S_0$, then its coefficients related to M_b are zero, and similarly if $s \in S_b$ then its coefficient related to M_0 are zero. Hence computing s from coefficient corresponding to points in M_0 (respectively, M_b) is equivalent to computing from M, and so M_0 and M_b are determining sets for S_0 and S_b, respectively. They are minimal determining sets because otherwise M would not be minimal. Clearly, stability and locality properties of M_0 and M_b are also inherited from M.

If M admits a stable splitting, then $S = S_0 + S_b$ and it is easy to see that

$$\{s_\xi\}_{\xi \in M} = \{s_\xi\}_{\xi \in M_0} \cup \{s_\xi\}_{\xi \in M_b}$$

is a stable splitting of the stable local basis $\{s_\xi\}_{\xi \in M}$.

4 Stable Splitting for Argyris Finite Element

Recall that the superspline subspaces $S_d^{r,\rho}(\triangle)$, $r \le \rho \le d$, of $S_d^r(\triangle)$ are defined as

$$S_d^{r,\rho}(\triangle) = \{\, s \in S_d^r(\triangle) : s \in C^\rho(v)\ \forall v \in V\}\,, \tag{9}$$

where V is the set of all vertices of $\triangle$.

Consider the Argyris finite element space obtained with $d = 5$, $r = 1$ and $\rho = 2$ in (9). Now for each $v \in V$, let T_v be any one of the triangles sharing the vertex v and let $M_v := D_2(v) \cap T_v$. For each edge e of the triangulation $\triangle$, let $T_e := \langle v_1, v_2, v_3\rangle$ be one of the triangles sharing the edge $e := \langle v_2, v_3\rangle$ and let $M_e := \left\{\xi_{122}^{T_e}\right\}$. Then from [8, Theorem 6.1] we have

Theorem 2. $\dim S_5^{1,2}(\triangle) = 6\#\{V\} + \#\{E\}$ *and*

$$M = \bigcup_{v \in V} M_v \cup \bigcup_{e \in E} M_e \tag{10}$$

is a stable local minimal determining set for $S_5^{1,2}(\triangle)$.

An example is given in Figure 1 (left).

4.1 Modified Argyris Space

We now modify the Argyris space to achieve the stable splitting. This construction is discussed in term of nodal basis functions in [6]. We will explain in Section 4.3 why this modification is required. Let us denote the modified Argyris space by $\tilde{S}$, where

$$\tilde{S} := \left\{s \in S_5^1(\triangle) : s \in C^2(v),\ \text{for all } \textit{interior} \text{ vertices } v \text{ of } \triangle\right\}. \tag{11}$$

Let us now differentiate between boundary vertices and interior vertices by using V_I and V_B for the sets of interior and boundary vertices respectively. And let E_I and E_B denote interior and boundary edges respectively, such that

$$V = V_I \cup V_B, \quad E = E_I \cup E_B.$$

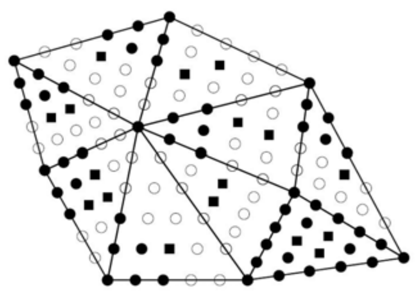


Fig. 1. Minimal determining sets for the Argyris space (*left*) and for the modified modified Argyris space (*right*). The points in the sets M_v, $\tilde{M}_v$ are marked by black dots, and those in M_e by black squares.

We describe a minimal determining set $\tilde{M}$ for this modified space $\tilde{S}$. Since we have modified the space only at the boundary vertices, so the points in M related to interior vertices and related to all edges, will belong to $\tilde{M}$. That is,

$$\left(\bigcup_{v \in V_I} M_v \cup \bigcup_{e \in E} M_e \right) \subset \tilde{M}.$$

However, we will have to modify the sets corresponding to the boundary vertices $v \in V_B$. First of all, we require that each T_v, $v \in V_B$, is a triangle sharing an edge with the boundary of Ω (we call it a *boundary triangle*). Furthermore, we add some more points to M_v, $v \in V_B$, as follows. Let us denote all edges of $\triangle$ emanating from a vertex $v \in V_B$, in counterclockwise order, by

$$E_v = \{e_1, e_2, \cdots, e_n\}.$$

Then clearly $e_1, e_n \in E_B$, and the triangle T_v is formed by either e_1, e_2 or e_{n-1}, e_n. For each e_i, let ξ_i be the (unique) domain point in $R_2(v) \cap e_i$, $i = 1, \ldots, n$. We set

$$\tilde{M}_v := M_v \cup \{\xi_1, \xi_2, \cdots, \xi_n\}.$$

Theorem 3. $\dim \tilde{S} = 6\#\{V_I\} + \#\{E\} + \sum_{v \in V_B} (4 + \#E_v)$ *and*

$$\tilde{M} := \bigcup_{v \in V_I} M_v \cup \bigcup_{e \in E} M_e \cup \bigcup_{v \in V_B} \tilde{M}_v. \tag{12}$$

is stable local MDS for modified Argyris space $\tilde{S}$.

Proof. We set the coefficients $\{c_\xi\}_{\xi \in \tilde{M}}$ for any spline $s \in \tilde{S}$ to arbitrary values and show that all other coefficients, i.e. $\{c_\xi\}_{\xi \in D_{5,\triangle} \setminus \tilde{M}}$, of s can be determined consistently.

Now first note that for each $v \in V_I$ and for each $e \in E$ the points in M_v and M_e are the same as for Argyris space. So we only need to prove that for each $v \in V_B$ the set $\tilde{M}_v$ is an MDS on $D_2(v)$. To this end, for each $v \in V_B$, we set the coefficients of s corresponding to points in $\tilde{M}_v$ and see that, in view of C^1 smoothness conditions, all coefficients corresponding to domain points in $D_2(v)$ can be determined consistently. Thus by [8, Theorem 5.15] $\tilde{M}$ is minimal determining set for the space $\tilde{S}$. Observe that $\tilde{M}$ is a stable MDS. Indeed, for each $v \in V_I$ and all edges $e \in E$ the stability follows from [8, Lemma 2.29]. And for each $v \in V_B$ the set $\tilde{M}_v$ is a stable MDS for S_5^1 on $D_2(v)$ by [8, Theorem 11.7]. Standard arguments show that $\tilde{M}$ is local. □

The minimal determining sets for Argyris space and for modified Argyris space over a small triangulation with nine triangles are illustrated in Figure 1.

4.2 Stable Splitting

Now we show how to determine a stable splitting $\tilde{M} = \tilde{M}_0 \cup \tilde{M}_b$ of the MDS $\tilde{M}$ for modified Argyris space $\tilde{S}$.

It is already understood that all those points of $\tilde{M}$ which are on the boundary will be in $\tilde{M}_b$ and those points lying in M_v, $v \in V_I$, and M_e along with the points in $R_2(v)$, $v \in V_B$, but not on either e_1 or e_n, will be in $\tilde{M}_0$. Consider, for each $v \in V_B$, the remaining point which lies in $R_1(v)$, $v \in V_B$, but not on the boundary edges. We denote this point by ξ_v. Whether ξ_v belongs to $\tilde{M}_0$ or $\tilde{M}_b = \tilde{M} \setminus \tilde{M}_0$ depends on the geometry of the boundary edges e_1 and e_n, as follows.

- If e_1 and e_n are non-collinear, then $\xi_v \in \tilde{M}_b$.
- If e_1 and e_n are collinear, then $\xi_v \in \tilde{M}_0$.

Indeed, in the non-collinear case the coefficient corresponding to ξ_v is zero for all $s \in \tilde{S}_0$, wheras in the collinear case it can be chosen freely. Figures 2 and 3 show points in $\tilde{M}_0$ and $\tilde{M}_b$ for the boundary vertex with collinear and non-collinear edges respectively.

Theorem 4. *$\tilde{M} = \tilde{M}_0 \cup \tilde{M}_b$ is stable splitting of MDS $\tilde{M}$.*

Proof. If $s \in \tilde{S}_0$, then all its Bézier coefficient on the boundary are zero since $s|_{\partial\Omega} = 0$. For those $v \in V_B$ where the boundary edges are non-collinear, the C^1 smoothness implies that the gradient at v is also zero, and hence the coefficient of s at ξ_v is also zero. This shows that $\tilde{S}_0 \subset \{s \in \tilde{S} : c_\xi = 0 \; \forall \xi \in \tilde{M}_b\}$. Conversely, assume $s \in \tilde{S}$ and $c_\xi = 0$ for all $\xi \in \tilde{M}_b$. Let $v \in V_B$ and $E_v = \{e_1, e_2, \cdots, e_n\}$ as before. Without loss of generality assume that $D_2(v) \cap e_1 \subset \tilde{M}_v$ and $R_2(v) \cap e_n \subset \tilde{M}_v$. Therefore $c_\xi = 0$ at all these points. However, due to the C^1 smoothness $c_\xi = 0$ also for the domain point in $R_1(v) \cap e_n$, both in the collinear and non-collinear case. This shows that $c_\xi = 0$ for all domain points on the boundary of Ω and hence $s|_{\partial\Omega} = 0$. Thus, $\tilde{S}_0 = \{s \in \tilde{S} : c_\xi = 0 \; \forall \xi \in \tilde{M}_b\}$, which completes the proof, see the discussion following Definition 1. □

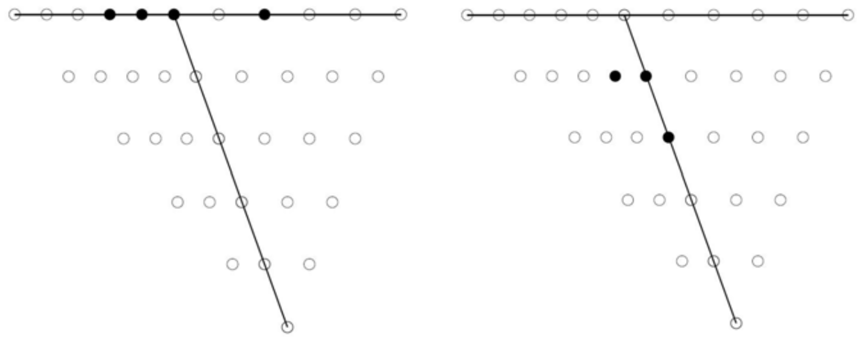


Fig. 2. Splitting of points in $\tilde{M}_v$, $v \in V_B$ for modified Argyris space with collinear boundary edges. *Left*: $\tilde{M}_v \cap \tilde{M}_b$, *right*: $\tilde{M}_v \cap \tilde{M}_0$.

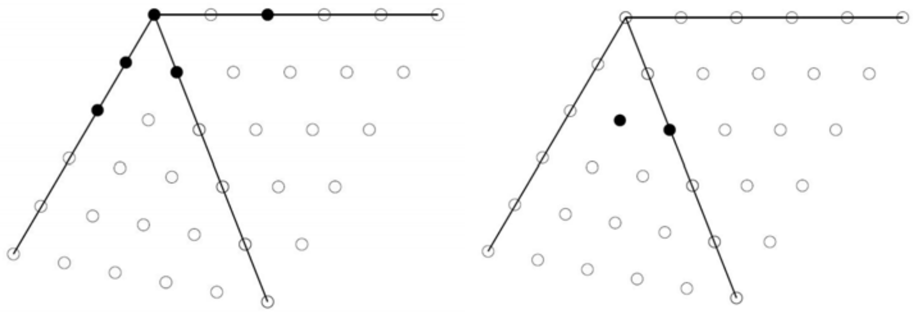


Fig. 3. Splitting of points in $\tilde{M}_v$, $v \in V_B$ for modified Argyris space with noncollinear boundary edges. *Left*: $\tilde{M}_v \cap \tilde{M}_b$, *right*: $\tilde{M}_v \cap \tilde{M}_0$.

4.3 Why Modification in Argyris Space Is Required

We now prove that modification is needed in Argyris space at the boundary vertices to achieve a stable splitting.

We first consider the Argyris space $S_5^{1,2}(\triangle)$ with M in Theorem 2 being its MDS, and show that no splitting $M = M_0 \cup M_b$ is possible in this case if there is a boundary vertex v with two triangles attached, and the boundary edges are non-collinear. On contrary, assume that such a splitting has been found. Let $T := \langle v_1, v_2, v_3\rangle$ and $\tilde{T} := \langle v_4, v_3, v_2\rangle$ be two triangles in $\triangle$ with v_3 as boundary vertex and assume that the edges $\langle v_3, v_4\rangle$ and $\langle v_3, v_1\rangle$ are boundary edges. Consider the set

$$M_{v_3} := D_2(v_3) \cap T = \{\xi_{005}, \xi_{014}, \xi_{023}, \xi_{104}, \xi_{113}, \xi_{203}\} \subset M,$$

see the Figure 4, and let

$$s|_T = \sum_{i+j+k=5} c_{ijk} B^5_{ijk}, \quad s|_{\tilde{T}} = \sum_{i+j+k=5} \tilde{c}_{ijk} \tilde{B}^5_{ijk},$$

where B^5_{ijk} and $\tilde{B}^5_{ijk}$ are Bernstein basis polynomials associated with T and $\tilde{T}$ respectively. In the case that the edges $\langle v_3, v_4\rangle$ and $\langle v_3, v_1\rangle$ are non-collinear, the points $\{\xi_{005}, \xi_{014}, \xi_{104}, \xi_{203}\}$ must be in M_b, because $s \in S$ has zero coefficients at these points. We show that $\{\xi_{113}, \xi_{023}\} \not\subset M_0$. Let (b_1, b_2, b_3) be barycentric coordinates of v_4 relative to T. Then by a C^2 smoothness condition, see [8, Theorem 2.28], across the edge $e := \langle v_3, v_2\rangle$ we can write

$$\tilde{c}_{230} = b_1^2 c_{203} + 2b_1b_2c_{113} + 2b_2b_3c_{014} + b_2^2 c_{023} + 2b_1b_3c_{104} + b_3^2 c_{005},$$

and because $\tilde{c}_{230} = c_{203} = c_{014} = c_{104} = c_{005} = 0$,

$$0 = 2b_1c_{113} + b_2c_{023},$$

which shows that c_{113} and c_{023} are linearly dependent so that ξ_{113}, ξ_{023} cannot be both in M_0. Moreover, we cannot shift one of these points to M_b because there is a spline $s \in S_0$ such that

$$c_{113}, c_{023} \neq 0,$$

e.g. s with $c_{113} = b_2$ and $c_{023} = -2b_1$. Note that $b_2 \neq 0$ if the boundary edges are non-collinear.

Moreover, we prove that no other MDS admits a stable splitting, either.

Theorem 5. *No MDS for the Argyris space can be stably split on arbitrary triangulations.*

Proof. Assume that the triangulation $\triangle$ is such that there is a boundary vertex v with two triangles T and $\tilde{T}$ attached, and the boundary edges are non-collinear at v, as in the above proof. Let M be some MDS for Argyris space.

From the dimension argument we know that there must be exactly six points in $M \cap D_2(v)$. For the non-collinear boundary edges, no points on boundary edges or in $R_1(v)$ can be in M_0 because, all the corresponding coefficients of splines in S_0 are zero. So the only candidates for M_0 are the points in $R_2(v)$ not on boundary edges. Now we discuss the relation between the coefficients $\tilde{c}_{131}, c_{113}, c_{023}$ of $s \in S_0$ at these points. By using C^1 and C^2 condition across the common edge of T and $\tilde{T}$ we get

$$\begin{aligned} \tilde{c}_{131} &= b_1c_{113} + b_2c_{023} \\ 0 &= 2b_1c_{113} + b_2c_{023} \end{aligned}$$

By subtracting these equations we can write

$$\tilde{c}_{131} = -b_1c_{113}$$

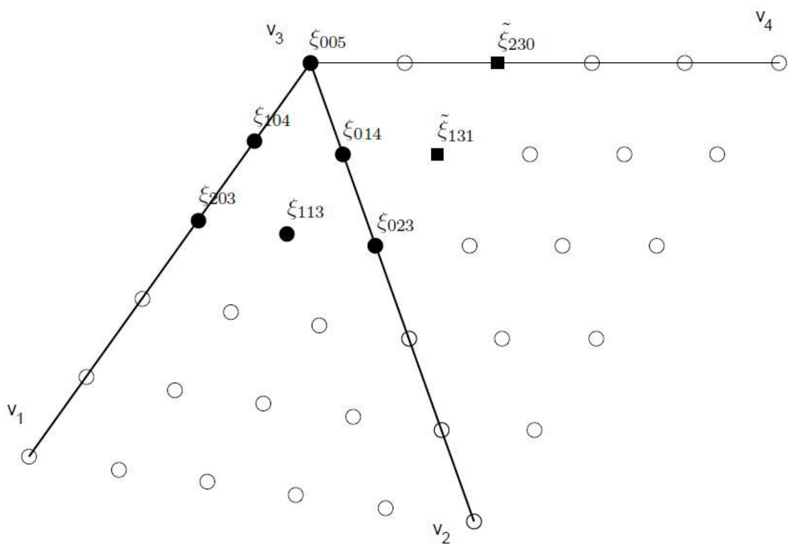


Fig. 4. The black dots are MDS points in $M_{v_3}, v_3 \in V_B$, for Argyris space. The two domain points marked by black squares are involved in the smoothness conditions discussed in the proof of Theorem 5.

Hence the three coefficients cannot be set arbitrarily. Only one of them can be chosen freely, which cannot be either $\tilde{c}_{131}$ or c_{113}. Indeed, let us choose e.g. c_{113} arbitrarily, then from the above equations we obtain

$$c_{023} = \frac{-2b_1 c_{113}}{b_2}$$

and hence $c_{023} \to \infty$ for $b_2 \to 0$ as the boundary edges get collinear. This would be unstable as the minimum angles in $T, \tilde{T}$ do not degenerate.

Thus ξ_{023} is the only point to be in M_0. It is easy to see that M_b must contain $\xi_{203}, \tilde{\xi}_{230}$ and three points in $D_1(v)$. Consider the basis spline s in S_b corresponding to $\tilde{\xi}_{230}$. Then its coefficient satisfy

$$\tilde{c}_{230} = 1, \quad c_{203} = c_{023} = 0, \quad c_\xi = 0, \quad \xi \in D_1(v)$$

Now again using C^1 and C^2 conditions we find

$$\tilde{c}_{230} = 2b_1 b_2 c_{113} \ \text{ or } \ c_{113} = \frac{1}{2b_1 b_2},$$

which is unbounded for $b_2 \to 0$ as the boundary gets flat. □

Remark 1. If a boundary vertex v has exactly two triangles attached and the boundary edges are not collinear at v, then stable splitting of an MDS is impossible for any spline space S where each spline is C^2 continuous at v. Indeed, this follows by the arguments in the proof of Theorem 5. In fact, it is easy to see that the set $D_2(v) \cap T$ as MDS for S on $D_2(v)$ cannot be split stably for a boundary vertex with any number of triangles attached.

5 C^1 Macro-Element Spaces

Now we discuss the possibility of stable splitting of minimal determining sets of some of the C^1 macro-element spaces.

5.1 Stable Splitting of Clough-Tocher Macro-Element Space

Given a triangulation $\triangle$ of a domain Ω, let $\triangle_{CT}$ be corresponding Clough-Tocher refinement of $\triangle$, where each triangle is split into three subtriangles, see Figure 5.

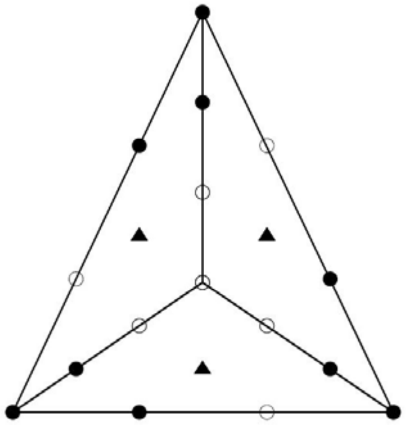


Fig. 5. A typical Clough-Tocher refinement of one triangle with points in M_v marked as black dots and points in M_e marked as black triangles

Consider the stable local MDS M given in [8, Theorem 6.5] for C^1 Clough-Tocher Macro-element space $S_3^1(\triangle_{CT})$ as

$$M = \bigcup_{v\in V} M_v \cup \bigcup_{e\in E} M_e, \tag{13}$$

where $M_v := D_1(v) \cap T_v$ and $M_e := \left\{\xi_{111}^{T_e}\right\}$, and T_v and T_e are triangles in $\triangle_{CT}$. Denote by V and E the sets of vertices and edges in $\triangle$, respectively. Let

$$S_0 := \left\{ s \in S_3^1(\triangle_{CT}) \; : \; s|_{\partial\Omega} = 0 \right\}.$$

Let V_I and V_B be the sets of interior and boundary vertices of $\triangle$, respectively. We assume that T_v is a boundary triangle for each M_v, $v \in V_B$. Then stable splitting for M is possible as follows. Clearly,

$$\left(\bigcup_{v \in V_I} M_v \cup \bigcup_{e \in E} M_e \right) \subset M_0. \tag{14}$$

However, M_0 may contain some more points from M_v, $v \in V_B$. Note that, for boundary vertices v, two points in M_v are always on the boundary and one is not. These two boundary points are in M_b but the point in M_v, which is not on the boundary, belongs to either M_0 or M_b depending on the geometry of boundary edges attached to v in the same way as the point ξ_v in Section 4.2. This point will be in M_0 for those boundary vertices where boundary edges are collinear. Otherwise it will be in M_b. Stability and locality follows as M is a stable local MDS for $S_3^1(\triangle_{CT})$.

5.2 Powell-Sabin Macro-Element Space

Now let for a given triangulation $\triangle$ of a domain Ω, $\triangle_{PS}$ be the corresponding Powell-Sabin refinement [8, Definition 4.18], see the Figure 6. For each $v \in V$, let T_v be some triangle of $\triangle_{PS}$ attached to v, and $M_v := D_1(v) \cap T_v$. Then

$$M = \bigcup_{v \in V} M_v \tag{15}$$

is a stable local minimal determining set for Powell-Sabin space $S_2^1(\triangle_{PS})$ [8, Theorem 6.9]. Now similarly if

$$S_0 := \left\{ s \in S_2^1(\triangle_{PS}) \ : \ s|_{\partial\Omega} = 0 \right\}$$

and if we take T_v to be a boundary triangle for M_v, $v \in V_B$, then M given in (15) for $S_2^1(\triangle_{PS})$ can be split stably in the same way as discussed above for the Clough-Tocher macro-element space.

5.3 Powell-Sabin-12 Macro-Element Space

Let $\triangle_{PS12}$ be the Powell-Sabin-12 refinement [8, Definition 4.21] of a given triangulation $\triangle$ of a domain Ω, see Figure 7. For each e of $\triangle$, let u_e be the midpoint of e and let v_T be the incenter of a triangle T in $\triangle$ attached to e. Let $\xi_e := \frac{v_T + u_e}{2}$ and $M_e := \{\xi_e\}$. For each vertex $v \in V$, let T_v be a triangle of $\triangle_{PS12}$ attached to v, and let $M_v := D_1(v) \cap T_v$. Then the set

$$M = \bigcup_{v \in V} M_v \cup \bigcup_{e \in E} M_e \tag{16}$$

is a stable local MDS for the space $S_2^1(\triangle_{PS12})$ [8, Theorem 6.13]. Now let

$$S_0 := \left\{ s \in S_2^1(\triangle_{PS12}) \ : \ s|_{\partial\Omega} = 0 \right\}.$$

Again, assuming that T_v is a boundary triangle of $\triangle_{PS12}$ for any bondary vertex v, we can split M into M_0 and M_b by the same method as for the Clough-Tocher elements. Then $M = M_0 \cup M_b$ is a stable splitting for $S_2^1(\triangle_{PS12})$.

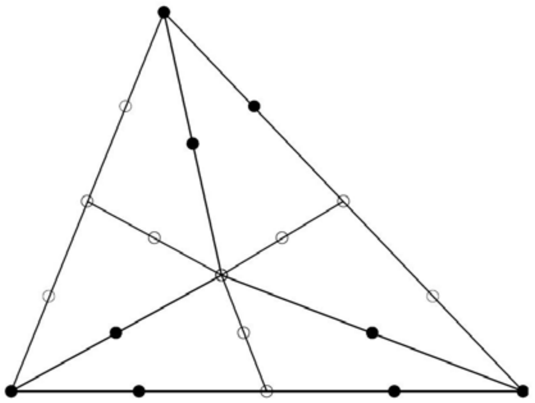


Fig. 6. Powell-Sabin refinement of one triangle with points in M_v marked as black dots

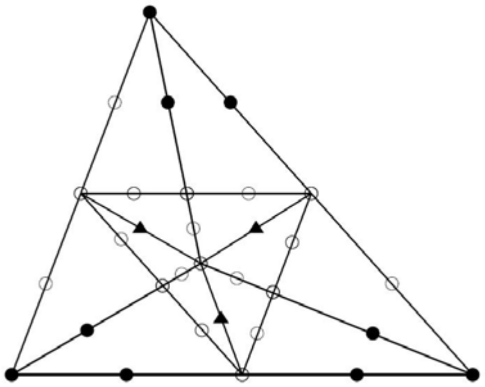


Fig. 7. A Powell-Sabin-12 refinement of one triangle with points in M_v marked as black dots and points in M_e marked as black triangles

5.4 Quadrilateral Macro-Element Space

Let $\diamondsuit$ be a strictly convex quadrangulation of a polygonal domain Ω and let $\triangle_Q$ be triangulation obtained by drawing in the diagonals of each quadriletral of $\diamondsuit$. Let V and E be the sets of vertices and edges of $\diamondsuit$. Here we will discuss the cubic spline space $S_3^1(\triangle_Q)$. Again let $M_v := D_1(v) \cap T_v$, for each $v \in V$, where T_v is a triangle in $\triangle_Q$ attached to v, and T_v is a boundary triangle in case of a boundary vertex v. For each $e \in E$, let T_e be some triangle in $\triangle_Q$ containing e and let $M_e := \left\{\xi_{111}^{T_e}\right\}$. Then

$$M = \bigcup_{v \in V} M_v \cup \bigcup_{e \in E} M_e \tag{17}$$

is a stable local MDS for the space $S_3^1(\triangle_Q)$ [8, Theorem 6.17]. Again the stable splitting of M for $S_3^1(\triangle_Q)$ is possible by the argument discussed above for other C^1 macro-elements.

Note that in [8, Section 6.5] the above triangle T_v is chosen such that it has the largest shape ratio $\operatorname{diam}(T)/\rho(T)$ among all triangles attached to v. This allows stable MDS even in the presence of small angles in $\triangle_Q$ if the smallest angle in $\diamondsuit$ is separated from zero. However, this choice of T_v might be unsuitable for stable splitting if v is a boundary vertex because we need T_v to be a boundary triangle whereas the shape ratio might be larger for some interior triangle attached to v. Therefore, our construction of stable splitting is valid only if $\triangle_Q$ satisfies the minimum angle condition.

References

1. Böhmer, K.: On finite element methods for fully nonlinear elliptic equations of second order. SIAM J. Numer. Anal. 46(3), 1212–1249 (2008)
2. Böhmer, K.: Numerical Methods for Nonlinear Elliptic Differential Equations: A Synopsis. Oxford University Press, Oxford (2010)
3. Courant, R., Hilbert, D.: Methods of Mathematical Physics, vol. II. Wiley Interscience, Hoboken (1989)
4. Dean, E.J., Glowinski, R.: Numerical methods for fully nonlinear elliptic equations of the Monge-Ampère type. Computer Methods in Applied Mechanics and Engineering 195, 1344–1386 (2006)
5. Davydov, O.: Stable local bases for multivariate spline spaces. J. Approx. Theory 111, 267–297 (2001)
6. Davydov, O.: Smooth finite elements and stable splitting, Berichte "Reihe Mathematik" der Philipps-Universität Marburg, 2007-4 (2007); An adapted version has appeared as Section 4.2.6 in [2]
7. Gilbarg, D., Trudinger, N.S.: Elliptic partial differential equations of second order. Springer, Berlin (2001)
8. Lai, M.J., Schumaker, L.L.: Spline Functions on Triangulations. Cambridge University Press, Cambridge (2007)

Mesh Segmentation and Model Extraction

Julie Digne[1], Jean-Michel Morel[1], Charyar Mehdi-Souzani[2], and Claire Lartigue[2]

[1] CMLA, ENS Cachan, CNRS, UniverSud,
61 Avenue du Président Wilson,
F-94230 Cachan

[2] LURPA, ENS Cachan, Univ. Paris Sud 11,
61 Avenue du Président Wilson,
F-94230 Cachan

Abstract. High precision laser scanners deliver virtual surfaces of industrial objects whose accuracy must be evaluated. But this requires the automatic detection of reliable components such as facets, cylindric and spherical parts, etc. The method described here finds automatically parts in the surface to which geometric primitives can be fitted. Knowing certain properties of the input object, this primitive fitting helps quantifying the precision of an acquisition process and of the scanned mires. The method combines mesh segmentation with model fitting. The mesh segmentation method is based on the level set tree of a scalar function defined on the mesh. The method is applied with the simplest available intrinsic scalar function on the mesh, the mean curvature. In a first stage a fast algorithm extracts the level sets of the scalar function. Adapting to meshes a well known method for extracting Maximally Stable Extremal Regions from the level set tree on digital images, the method segments automatically the mesh into smooth parts separated by high curvature regions (the edges). This segmentation is followed by a model selection on each part permitting to fit planes, cylinders and spheres and to quantify the overall accuracy of the acquisition process.

Keywords: Mesh segmentation, model fitting, mire accuracy.

1 Introduction

Laser scanners acquire object surfaces with growing accuracy. The question arises of evaluating quickly and reliably this accuracy. This paper proposes an approach to perform this evaluation automatically, based on mesh segmentation and model fitting, without the use of calibrated mires.

The raw output of a laser scanner is a set of samples on the acquired surface. Building a mesh from this set of points can be done (e.g.) by the classic methods in [18], [8], [14], [3]. The evaluation of the precision of the acquisition pipeline must be based on parts of scanned objects that have homogeneous curvatures, which requires a mesh segmentation into high and low curvature parts. The low curvature parts are easier to parameterize by polynomial models such as planes, cylinders, spheres, etc. and permit to perform a model selection. The goal is not necessarily to find a global model of the shape (which is often impossible), but to find models that significantly fit some parts of the data. The global evaluation of the accuracy must be based on these parts. In particular one goal of the analysis proposed here is to evaluate sharp edges positions for mires of mechanical pieces.

The remainder of this paper is divided as follows. Section 2 reviews the current state of the art for mesh segmentation and model regression. Section 3 explains the segmentation method used in this paper, section 4 presents the model fitting and model selection choices and finally section 5 presents numerical results.

2 Previous Work

Mesh segmentation has been widely studied and there are extensive segmentation technique reviews, [4], [26] and [1].

Segmentation methods can be divided into two categories: semantic segmentation methods and geometric segmentation methods. Semantic segmentation methods try to segment the mesh into meaningful parts. Those methods include [28], where the clusters have to exhibit symmetries, and [17], where the mesh is segmented into core components and protrusions. [29] finds the skeleton of a shape, identifies junction areas and uses them to get a semantic oriented segmentation. Its goal can be to perform shape morphing ([27]). On the contrary, geometric segmentation methods aim at dividing the mesh into clusters that have similar geometric properties (constant curvature, for example). These methods are the ones we are interested in. Indeed, segmenting the mesh into homogeneous curvature parts is the only way to fit models to the clusters.

Mesh clustering techniques have made intensive use of the k-means algorithm, trying to group facets in clusters with it. For example, [33] clusters the mesh by performing a k-means on the normals to the mesh. Region merging and region growing are then used to get the final classification. The k-means algorithm is also used in [35] where vertices are clustered according to their curvatures. [19] extends work on feature sensitive remeshing techniques to generate a mesh that is suited for hierarchical mesh clustering and then uses the k-means algorithm to segment the triangles. It assigns triangles to clusters according to the distances from an iteratively updated representative triangle. In [20], the segmentation is based on curvature tensor field analysis. The patches exhibit a nearly constant curvature. The k-means classification is used to classify vertices according to their principal curvatures. A region growing extracts triangle regions from vertex curvature information by taking sharp edges into account. Regions are then merged and boundaries are finally rectified by removing discontinuities. The segmentation between homogeneous Gaussian curvature parts was also considered in [32] for building a better surface parmeterization.

In [16], another type of segmentation is performed: it is based on the mesh dual graph. This dual graph is simply built by considering each triangle as a dual node and linking the dual nodes if the corresponding faces are adjacent on the surface. In this dual graph contracting an edge means grouping two faces into a single cluster. Therefore, by defining an adequate edge contraction cost, the mesh segmentation is obtained by minimizing the edge contraction cost to find the best k clusters (k being a user-specified parameter).

Based on this method, a mesh segmentation interleaved with a model regression has been introduced in [5]. Simple primitives are considered for model fitting (plane, sphere and cylinder primitives). The method described in [16] is modified by changing the edge contraction cost: the cost of contracting the edge is the error to the model best fitting the set of points included in both clusters. Our proposed segmentation method is completely different from this last method, but we also chose to study only planes, cylinders and spheres, simply because these are very simple primitives that are widely used for mechanical objects. They seem to be realistically the only ones for which a reliable model decision can be made.

An interesting method was proposed in [22]: the watershed segmentation, coming directly from the 2D image processing field was adapted to the mesh segmentation problem. The watershed segmentation was also studied in [36]. The idea of adapting the techniques of 2D image processing was also used in [13] with a completely different method: the image level set selection tools and the Maximally Stable Extremal Region method ([6], [23]). In this paper, we will use this method to segment the meshes (see section 3).

The idea of selecting level sets for segmenting a mesh was proposed also in [34] in a completely different context. The shape is decomposed into a smooth base and a height function defined over the mesh. Then a level of the height function is selected and the corresponding level set is extracted. The main difference of the method proposed here lies in the definition of the height function and the fact that only one level was selected in this method while here the threshold is automatic and adaptive.

Model regression for mesh parts has also generated a considerable literature, mostly for fitting quadrics ([24], [2], [9], [31], [11]) or NURBS ([21], [25]). Much less work has been done for fitting primitives to point sets. [30] and [10] are notable exceptions.

The next section describes a mesh segmentation method.

3 Mesh Segmentation

The mesh segmentation tool used in this paper was introduced in [13]. It is based on ideas coming from 2D image processing. An image is entirely represented by its level set tree (see Fig. 1). This tree is built by considering upper level sets of the gray values, the sets $u^\lambda = \{x \in \Omega | u(x) \geq \lambda\}$ where u is an image defined over Ω, a bounded subset of $\mathbb{R}^2$. Then, one has $u^{\lambda'} \subset u^\lambda$ as soon as $\lambda' \geq \lambda$ and, if we consider a connected component $C^{\lambda'}$ of the level set $u^{\lambda'}$, then there is a connected component C^λ of the level set u^λ such that $C^{\lambda'} \subset C^\lambda$. For a quantified image, the tree is built by adding a node per level set connected component and adding a parent-child relationship between nodes A and B when the corresponding connected components satisfy $A \subset B$. The root of this tree is

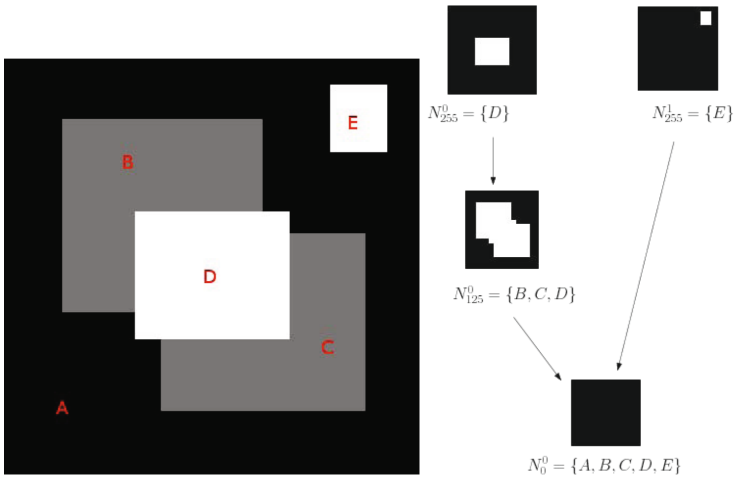


Fig. 1. A gray color image (left) and its level set tree (right)

the level set with smallest quantization value (usually 0 for gray-scale images). Notice that the defined level set tree uses upper level sets. A similar construction can be done for lower level sets.

This tree is then used to select meaningful level sets. In [23], selected level set connected components correspond, in a nutshell, to those that least move when perturbing slightly the level. More precisely, for each level set connected component, the *area change rate* $q(C^\lambda)$ is

$$q(C^\lambda) = \frac{A(C^{\lambda+\delta}) - A(C^{\lambda-\delta})}{A(C^\lambda)}$$

where $C^{\lambda\pm\delta}$ is the connected component with level $\lambda\pm\delta$ containing (or contained in) C^λ, and $A(C^\lambda)$ is the area of C^λ. *Maximally Stable Extremal Regions* (MSER) are local minima in the level set tree of the area change rate. Computing and comparing the area change rates is very easy using the level set tree structure. In [13] this parameterless segmentation method was adapted to functions defined over meshes.

3.1 Defining the Level Set Tree on Manifold Meshes

Going from the pixel image to a function defined on a mesh is simple: triangles play the role of pixels and two triangles will be said adjacent if and only if they share an edge. If the function is defined on triangles (i.e., the function is constant over each triangle), then no additional work is needed. On the contrary if the functions is defined only on the vertices then a value should be deduced for each triangle. To do that one can either take the barycentric, minimum or the maximum value depending on the purpose of the tree.

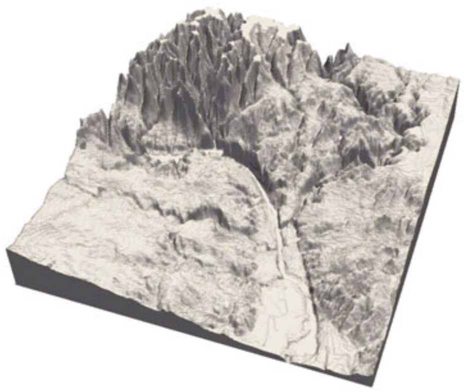


Fig. 2. Rendering of a digital elevation model

All the definitions then follow naturally from the 2D case. The only condition for the method to work on meshes is that each edge should be adjacent to at most 2 triangles. A fast algorithm was proposed in [13] for building the level set tree while storing the areas of each node. Once the tree is built, computing the area change rate for each node is obvious (simply go up or down in the tree until reaching nodes with levels $\lambda \pm \delta$) and this yields a MSER algorithm for meshes.

As explained in [13], the function should be quantized. In fact the whole method relies on two parameters once the function is chosen: the quantization step, and the derivative step δ. In the experiments of this paper we always chose those steps to be equal. The number of quantization steps is a value between 20 and 50.

3.2 Choosing the Adequate Scalar Function on a Mesh

At this point, given a function defined on a mesh, there is a method to segment the mesh into regions that are homogeneous for this function. But we need a suitable function for primitive fitting. The function choice is very dependent on the goal of the segmentation or the type of mesh. For example, on Fig. 2, a digital elevation model is rendered. If one wants to segment this model then the obvious function to use is the height of the vertices. But then we get closer to the image segmentation problem since there are two ways of considering the data: either as a gray-valued image using 2D-MSER, or as a set of grid points with height values giving a set of 3D coordinates with a mesh structure. A height value for each triangle can be deduced from the height values of all triangle vertices: here, the minimum of the vertices values was used. With this choice the mesh process ends up using a much finer information, since it adds precise triangle areas (which depend on the height) instead of adding a constant 1 area to compute the node areas. Fig. 3 compares applying 2D-MSER to the gray-level image or using Mesh-MSER on the corresponding mesh. Notice that the river is much better detected by Mesh-MSER than by the classic 2D MSER.

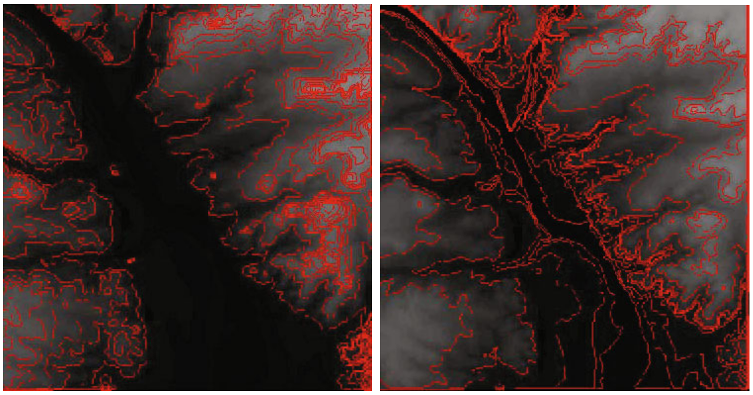


Fig. 3. 2D-MSER on the gray-level image (left) and Mesh-MSER applied to the corresponding mesh (right)

Following this idea of a height function defined on a plane, we shall define the height function as the distance between a vertex initial position and the vertex position after an intrinsic mesh smoothing. This way the mesh can be described as a smooth base with its normals, together with a scalar elevation function defined over the smooth base (Fig 4). The smoothing is done by applying the intrinsic heat equation (mean curvature motion) and the height function is nothing but the mean curvature.

Computing the mean curvature motion for surfaces is a very hard problem, especially when we are dealing with non smooth data (raw data coming directly from the laser scanner). In [14], it was proven that this motion can be approximated by projecting each point onto is local regression plane. We use this simple formulation here to build the height function.

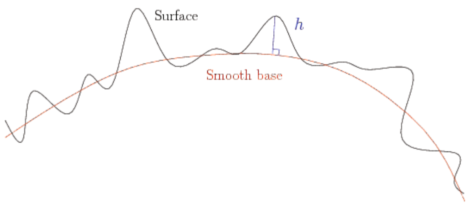


Fig. 4. "base+height" function

This idea is close to the method described in [34]. Fig. 5 compares the clustering obtained on a simple object using the same height function. But [34] selects globally one single level, and risks therefore creating unstable cluster boundaries. On the other hand Mesh-MSER tends to over-segment the edges, but extracts nicely the large homogeneous parts.

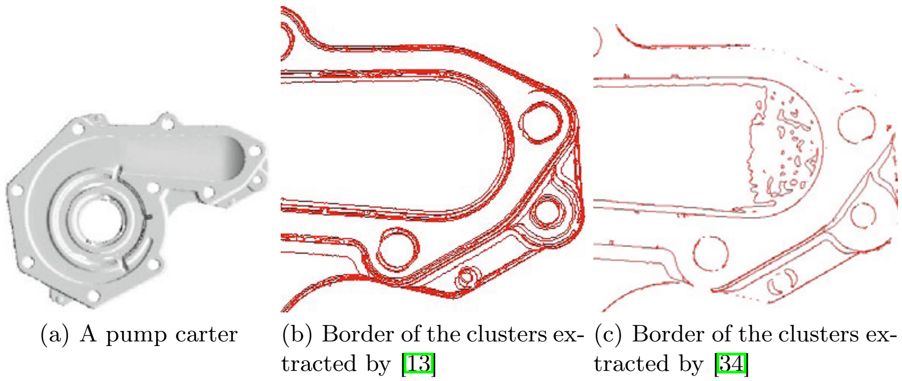

(a) A pump carter (b) Border of the clusters extracted by [13] (c) Border of the clusters extracted by [34]

Fig. 5. Comparison between the borders of the levels extracted by Mesh-MSER [13] and [34]

Naturally, the mean curvature is not the only possible function choice: depending on the goal of the segmentation one could use another function. If the goal is a geometric segmentation, then the function should contain geometric information. This explains why the mean curvature is an obvious choice: it is the simplest and lowest local isotropic derivative on the manifold. The Gaussian curvature or the principal curvatures are obvious alternatives.

The proposed mesh segmentation method works for non geometric surfaces (see [13] for examples of non geometric models segmentation or the experiment of fig. 2). Yet in this paper our goal is to extract geometric primitives to validate the accuracy with which the shape was built (or alternatively the accuracy of the scanner, given an accurate 3D pattern.) Thus that the experiments will only consider mechanical or geometric shapes.

4 Model Regression

This section proposes simple methods to estimate models (planes, cylinders and spheres) from a set of points. Those methods are robust to partial shapes. In other terms they work with half spheres or cylinder parts as well. The results of the direct method were compared with those obtained when the method is complemented by RANSAC ([15]). No substantial accuracy gain was noticed using RANSAC, so that the final numerical examples do not use it.

Fitting a plane model to a set of samples is easy by computing the point set barycenter and centered covariance matrix. The principal component analysis (PCA) of the matrix yields the normal to the fitted plane: it is the eigenvector corresponding to the least eigenvalue. Cylinder and sphere fitting is a little more complex. In the remainder of this section $(p_i)_{i=1\cdots N}$ will be a set of N points on which models will be tested. The normals $\mathbf{n_i}$ are calculated for all points p_i by local planar regression.

4.1 Cylinder Extraction

Fitting a cylinder to point sets has been studied in [5], [10] and [30]. The approach proposed here is slightly different. If the samples covered the full radial range, then a simple barycenter computation would yield a point on the principal axis. Yet this estimation does not work when the samples cover only a part of the cylinder, or when, as usual with raw data, the cylinder is not uniformly sampled. Given a set of points, one can find the degree 2 fitting polynomial and deduce the principal curvatures k_1 and k_2 for each point p_i. If the points were actually sampled on a cylinder with radius r then one of their principal curvatures would be 0 and the other would be $\pm\frac{1}{r}$ (the sign depending on the surface orientation). Denote by k_i the principal curvature of p_i with largest absolute value, then point $q_i = p_i - \frac{1}{k_i} \cdot \mathbf{n_i}$ must lie on the cylinder axis. In practice for a cylinder this procedure gives an elongated cluster of points. Computing the barycenter and first principal direction of those points yields the axis direction. Knowing the axis position and direction, the cylinder radius is estimated as the mean distance between the points and the axis. With this updated radius r positions of the q_i can be also updated:

$$q_i = \begin{cases} p_i - r \cdot \mathbf{n_i} \text{ if } \mathbf{k_i} > \mathbf{0} \\ p_i + r \cdot \mathbf{n_i} \text{ if } \mathbf{k_i} \leq \mathbf{0}. \end{cases}$$

Notice that for cylinders all k_i have the same sign as soon as the normals are precise enough. The updated points q_i allow for the update of the axis position and directions and the process can be iterated. For clarity, the process is summed up in Algorithm 1.

Algorithm 1. Fitting a cylinder to a set of points

```
Data: An input set of points $p_i$ with oriented normals $\mathbf{n_i}$, a number of
      iterations $K$
Result: A point $O$ and a direction $\mathbf{v}$ defining the cylinder axis and the radius $r$
1 Compute the principal curvatures by local polynomial regression ;
2 Compute $k_i = argmax(|k_1|, |k_2|)$ for all $p_i$;
3 for $i = 1 \cdots K$ do
4     Compute $q_i = p_i - \frac{1}{k_i} \cdot \mathbf{n_i}$;
5     Compute the barycenter $O$ and the principal direction $\mathbf{v}$ of the set $q_i$ by
      PCA;
6     Compute $r = \frac{1}{N} \sum_{i=1 \cdots N} dist(p_i, (O, \mathbf{v}))$;
7     Update $k_i \leftarrow sign(k_i)\frac{1}{r}$;
```

4.2 Sphere Extraction

The sphere case is very similar to the cylinder case. The parameters to estimate are simply the center and the radius of the sphere. As before, because points can cover only a small part of the sphere, the barycenter of the points is not an estimate for the center position. But points lying on a sphere have principal curvatures $k_1 = k_2 = \pm\frac{1}{r}$. Denote by k_i the estimated mean curvature $\frac{k_1+k_2}{2}$. Then points $q_i = p_i - \frac{1}{k_i} \cdot \mathbf{n_i}$ would ideally be all located at the exact sphere center. In practice, they form a cluster of points around the sphere center. Computing the barycenter of this cluster yields an estimated sphere center O. Knowing the sphere center, the new sphere radius r can be estimated as the mean of the distances $|Op_i|$. Using this new radius, the q_i can be updated:

$$q_i = \begin{cases} p_i - r \cdot \mathbf{n_i} \text{ if } \mathbf{k_i > 0} \\ p_i + r \cdot \mathbf{n_i} \text{ if } \mathbf{k_i \leq 0} \end{cases}$$

The sign of k_i is the same on the whole point set for real sphere surfaces. The updated points q_i allow for the updating of the center position and the process can be iterated. For clarity, the process is summed up in Algorithm 2.

Algorithm 2. Fitting a sphere to a set of points

```
  Data: An input set of points $p_i$ with oriented normals $\mathbf{n_i}$, a number of
        iterations $K$
  Result: The center $O$ of the sphere and its radius $r$
1 Compute the principal curvatures and normal by polynomial regression ;
2 Compute $k_i = (k_1 + k_2)/2$ for all $p_i$;
3 for $i = 1 \cdots K$ do
4     Compute $q_i = p_i - \frac{1}{k_i} \cdot \mathbf{n_i}$;
5     Compute the barycenter $O$ of the set $q_i$ by PCA;
6     Compute $r = \frac{1}{N} \sum_{i=1 \cdots N} dist(p_i, O)$;
7     Update $k_i \leftarrow sign(k_i)\frac{1}{r}$;
```

4.3 Model Selection

Using the above described techniques, one can estimate model parameters from point sets. Yet, one has to select the model that actually best fits the data and also of course discard regions which are neither plane, nor cylindrical or spherical. Here we shall use an apparently naive approach, but which turns out to be the correct one. For each vertices cluster, all three models are fitted and the root mean square error of the model fitting is computed. Only the model corresponding to the least root mean square error is considered as a potential model. To see if this model really fits the data the error standard deviation will be compared to the noise standard variation, which therefore has to be estimated.

Following the scale space strategy developed in [14], we assume that the noise is represented by the motion amplitude created by applying the projection filter (projecting the point onto its local regression plane). This estimation could introduce a small bias, since it was proven that this filter is tangent to the mean curvature motion. But on the large regions of the mesh the surface curvature is actually very small with respect to the noise curvature, thus this bias is negligible (of the order of 1%).The estimated standard deviation σ_b of the estimated noise

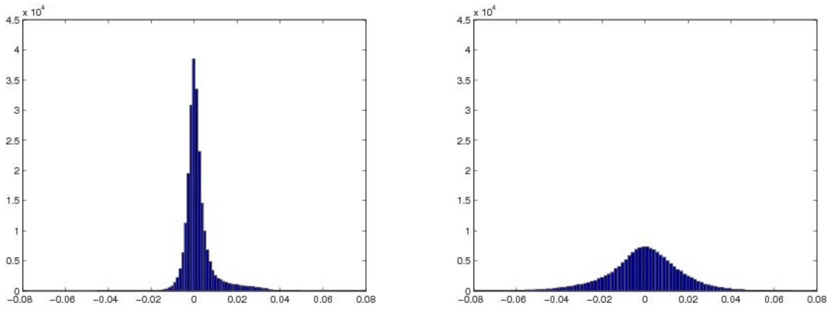


Fig. 6. Histograms for the estimated noise (left) and the model fitting error (right) on the diamond shape

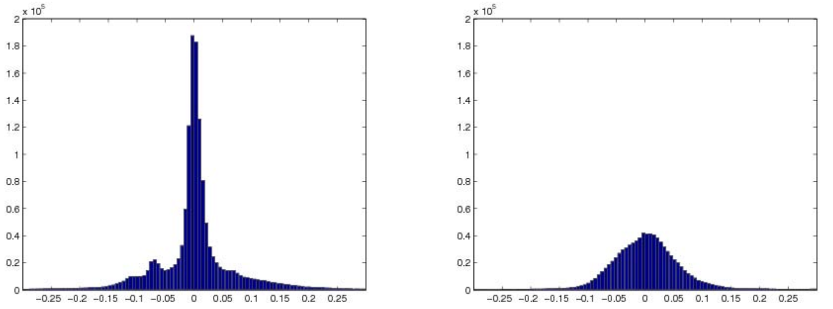


Fig. 7. Histograms for the estimated noise (left) and the model fitting error (right) on the first mire

yields a threshold for a potential model acceptance. If the fitting error for the best model (among the three models considered here) is less than $\lambda\sigma_b$, the model is accepted, otherwise the cluster is said to have no model. The parameter λ is necessarily empirical and has to be chosen by the user, because it corresponds to the deformation of the ideal model caused by the scanner (or present in the object itself). In all our experiments with a high precision scanner (20μ nominal precision) and assumedly very accurate industrial mires, we chose $\lambda = 4$, thus allowing for a bias proportional to the noise.

Figs 6 and 7 show the histograms of the estimated noise and fitting error. The fitting error is not Gaussian for multiple reasons: first, multiple scans cover each of the clusters and each of these scans is acquired using a different laser orientation. Each acquisition orientation generates a Gaussian noise, which may not have the same characteristics than for another orientation, yielding a mixture of gaussians for the global noise. Second, a powder had to be added to the metallic mires before the scanning process. This additional rugosity is a kind of noise which is not necessarily Gaussian.

The above strategy is very simple, and one might wonder why no better model selection tool was used. For example one could have used the Minimum Description Length (MDL) theory [7] or the Number of False Alarms (NFA) theory [12]. For one thing, in our case we considered only models with a very small number of parameters. Thus the huge number of samples in the raw data (several millions) causes the standard variation alone to decide the model selection result given by either MDL or NFA. In a case with a simple model and an overwhelming number of samples, both methods come down to the comparison of the root mean square error for each model. Indeed, both the MDL length and the $\log NFA$ have the order of magnitude $N \log \sigma_M$, where N is the number of samples and σ_M the observed root mean square error for the model M. The other terms are negligible. Thus the best MDL and the lowest NFA are given by comparing σ_M for the different choices of M. In both frameworks, the number of bytes used to describe the model will always be negligible compared to the number of bytes used to encode the offsets (proportional to the root mean square error). This justifies the “naive” approach.

5 Numerical Results

In this section we present segmentation and model regression results on two different kinds of industrial mires. We start with synthetic CAD meshes (Figs 8, 9 and 10). In such a setting the segmentation results are of course very clean, and can easily be compared to state of the art segmentation methods (see the results obtained for the fandisk, compared to other methods). The method was tested on several industrial mires scanned by a triangulation laser scanner with multiple scans (Figs 11,12, 13, 14 and 15). There is a serious acquisition noise as can be seen on the mesh renderings, due to the fact that an additional powder had to be spread on the metallic objects before the acquisition to make the scanner work.

5.1 Finding a Global Model

Since the points are clustered, and a model is found whenever possible on the data, a global model can be found for the whole object as soon as every (non-edge) class has a corresponding model. Consider for example the diamond shape used in this paper. It is composed of only planar parts, with the exception of the edge class. The equations of the planes are also known by plane fitting. By finding the intersections of planes corresponding to neighboring connected components one can recover the set of vertices defining the shape entirely. Those vertices are found as the intersections of either 3 or 4 planes (in the 4 planes case, the intersection is found as the point minimizing the distance to all 4 planes). Vertices are then linked, yielding the result of figure 16. It gives an evaluation of the whole acquisition process. Indeed, the diamond shape is supposed to be a perfect geometric object: for example, edges of the upper octogon must have same length. To quantify this, we sorted edges into classes (upper octogon, middle octogon, lower octogon) and computed the mean length of each class, the

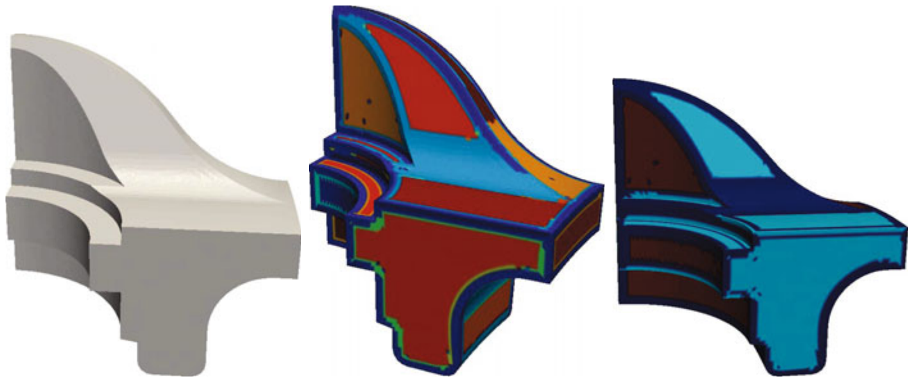

(a) Fandisk model (b) Clusters found by mesh segmentation (c) Classification

Fig. 8. Fandisk model. Fig. 8(b) shows the clusters plotted in random colors. The classification 8(c) shows plane clusters in cyan, cylinder clusters in red and spheres in yellow, classes without models are drawn in red. The fandisk model is provided courtesy of MPII by the AIM@SHAPE Shape Repository.

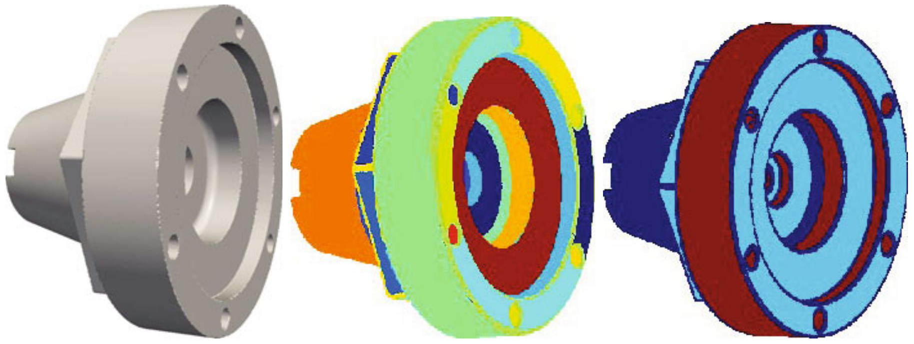

(a) Mesh (b) Clusters found by mesh segmentation (c) Classification

Fig. 9. A synthetic mesh (left); clusters (middle); models (right). Fig 9(b) plots clusters in random colors. Fig 9(c) shows plane clusters in cyan, cylinder clusters in red and spheres in yellow, classes without models are drawn in dark blue.

standard deviation of the length (measured in millimeter) compared to the class mean length is 0.0958. The error is therefore close to $100 \mu m$. It can have three distinct explanations: default of the mire (unlikely), variable thickness due to the additional powder spread on the object before scanning, and calibration default, which is the most likely explanation.

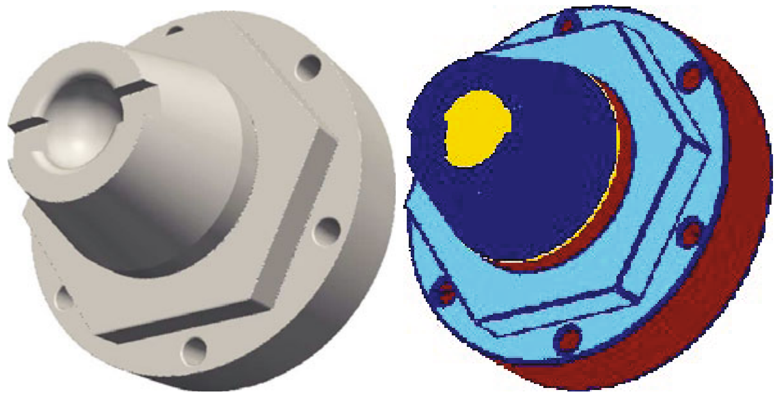

(a) Mesh (b) Classification

Fig. 10. A synthetic mesh (left); found models (right). Fig 10(b) shows plane clusters in cyan, cylinder clusters in red and spheres in yellow, classes without models are drawn in dark blue.

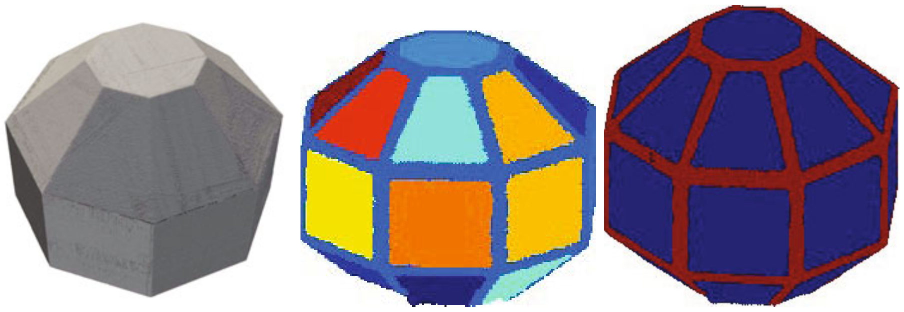

(a) Acquired shape mesh (b) Clusters found by mesh segmentation (c) Classification

Fig. 11. A simple shape formed only by planes. The classification 11(c) shows plane clusters in blue and edge classes in red.

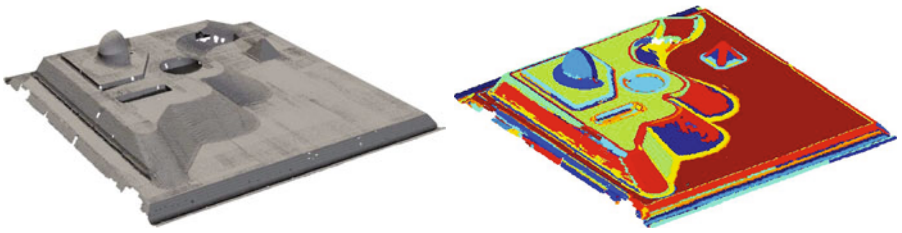

(a) Mesh of Mire 1 (b) Clusters found for the first mire

Fig. 12. Clusters found on the first mire. Clusters are given a random color.

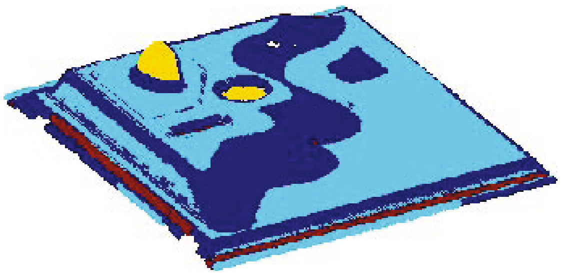


Fig. 13. Model selection (dark blue: no model; cyan: plane; yellow: sphere; red: cylinder)

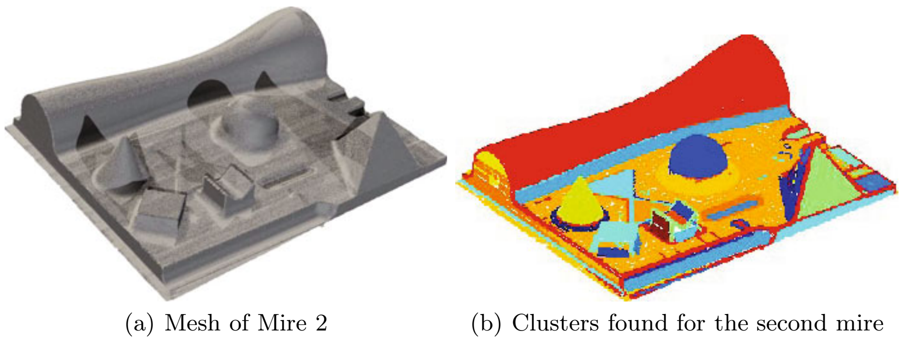


Fig. 14. Clusters found on the second mire. Clusters are given a random color.

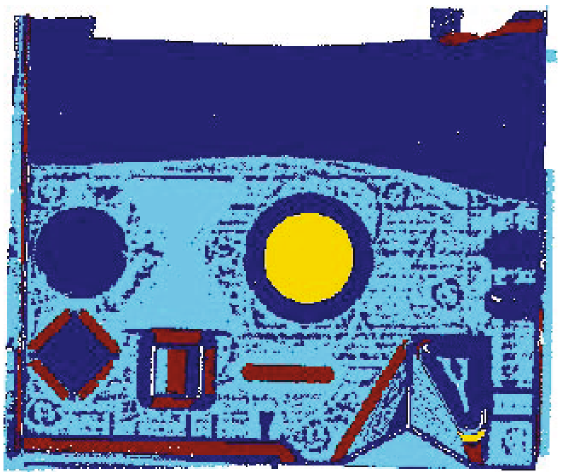


Fig. 15. Model selection (dark blue: no model; cyan: plane; yellow: sphere; red: cylinder)

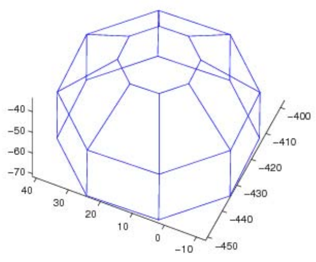


Fig. 16. Edges found by plane intersections on the diamond shape

6 Conclusion and Future Work

In this paper, we demonstrated that the framework defined by the level set tree yielded a mesh segmentation that made the model fitting step possible. Thanks to the model fitting, models can be estimated and compared to the theoretical properties of the shape. This can lead to a fair evaluation of the whole design-engineering-acquisition loop. Future work will focus on developing a cross-validation technique to distinguish between the three sources of errors mentioned, using multiple scans of the same mire under varying orientation.

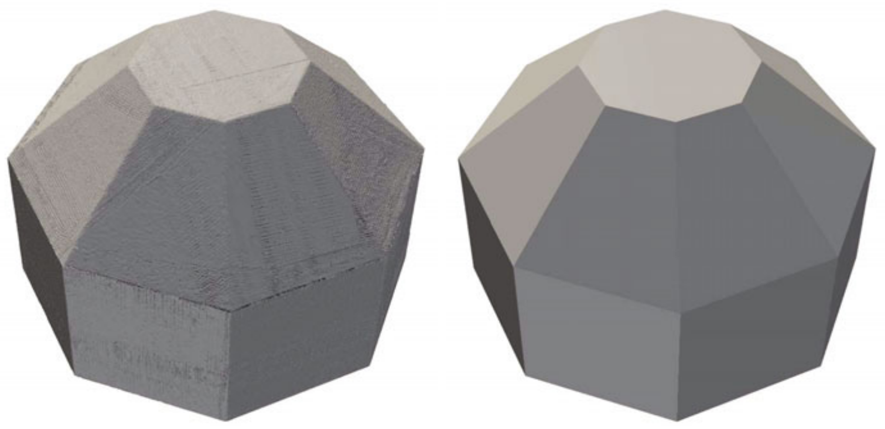


Fig. 17. The scanned shape (left) containing $200k$ vertices and the obtained model (right) containing only 25 vertices

Acknowledgements. The authors acknowledge support by DGA, French Ministry of Defense, by the ERC advanced grant "Twelve labours", the Centre National d'Etudes Spatiales, and by the Office of Naval Research (grant N00014-97-1-0839).

References

1. Agathos, E., Pratikakis, I., Perantonis, S., Sapidis, N., Azariadis, P.: 3d mesh segmentation methodologies for cad applications. Comput. Aided Des. Appl. 4(6), 827–841 (2007)
2. Al-Subaihi, I., Watson, G.A.: Algebraic fitting of quadric surfaces to data. Communications in Applied Analysis 9(3/4), 539–548 (2005)
3. Amenta, N., Bern, M.: Surface reconstruction by Voronoi filtering. In: SCG 1998: Proceedings of the Fourteenth Annual Symposium on Computational Geometry, pp. 39–48. ACM, USA (1998)
4. Attene, M., Katz, S., Mortara, M., Patane, G., Spagnuolo, M., Tal, A.: Mesh segmentation - a comparative study. In: SMI 2006, p. 7. IEEE Computer Society, Washington, DC, USA (2006)
5. Attene, M., Falcidieno, B., Spagnuolo, M.: Hierarchical mesh segmentation based on fitting primitives. Vis. Comput. 22, 181–193 (2006)
6. Ballester, C., Caselles, V., Monasse, P.: The tree of shapes of an image. Tech. rep. (2001)
7. Barron, A., Rissanen, J., Yu, B.: The minimum description length principle in coding and modeling (invited paper), pp. 699–716. IEEE Press, Piscataway (2000)
8. Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., Taubin, G.: The ball-pivoting algorithm for surface reconstruction. IEEE TVCG 5, 349–359 (1999)
9. Cao, X., Shrikhande, N., Hu, G.: Approximate orthogonal distance regression method for fitting quadric surfaces to range data. Pattern Recogn. Lett. 15(8), 781–796 (1994)
10. Chaperon, T., Goulette, F.: Extracting cylinders in full 3d data using a random sampling method and the gaussian image. In: Proceedings of the Vision Modeling and Visualization Conference, VMV 2001, Aka GmbH, pp. 35–42 (2001)
11. Dai, M., Newman, T.S.: Hyperbolic and parabolic quadric surface fitting algorithms - comparison between the least squares approach and the parameter optimization approach. Tech. Rep. TR-UAH-CS-1998-02 (1998)
12. Desolneux, A., Moisan, L., Morel, J.: From Gestalt Theory to Image Analysis: A Probabilistic Approach. Springer, Heidelberg (2008)
13. Digne, J., Morel, J.M., Audfray, N., Mehdi-Souzani, C.: The level set tree on meshes. In: Proceedings of the Fifth International Symposium on. 3D Data Processing, Visualization and Transmission, Paris, France (May 2010)
14. Digne, J., Morel, J.M., Souzani, C.M., Lartigue, C.: Scale space meshing of raw data point sets. Computer Graphics Forum (2011)
15. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24(6), 381–395 (1981)
16. Garland, M., Willmott, A., Heckbert, P.S.: Hierarchical face clustering on polygonal surfaces. In: Proceedings of the 2001 Symposium on Interactive 3D Graphics, I3D 2001, pp. 49–58. ACM, USA (2001)

17. Katz, S., Leifman, G., Tal, A.: Mesh segmentation using feature point and core extraction. The Visual Computer 21(8-10), 649–658 (2005)
18. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: SGP 2006, Eurographics, Switzerland, pp. 61–70 (2006)
19. Lai, Y.K., Zhou, Q.Y., Hu, S.M., Martin, R.R.: Feature sensitive mesh segmentation. In: Proceedings of the 2006 ACM Symposium on Solid and Physical Modeling, SPM 2006, pp. 17–25. ACM, USA (2006)
20. Lavoué, G., Dupont, F., Baskurt, A.: A new cad mesh segmentation method, based on curvature tensor analysis. Comput. Aided Des. 37, 975–987 (2005)
21. Ma, W., Kruth, J.P.: Nurbs curve and surface fitting for reverse engineering. The International Journal of Advanced Manufacturing Technology 14, 918–927 (1998)
22. Mangan, A.P., Whitaker, R.T.: Partitioning 3d surface meshes using watershed segmentation. IEEE TVCG 5(4), 308–321 (1999)
23. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In: British Machine Vision Conference, vol. 1, pp. 384–393 (2002)
24. Petitjean, S.: A survey of methods for recovering quadrics in triangle meshes. ACM Comput. Surv. 34(2), 211–262 (2002)
25. Piegl, L., Tiller, W.: The NURBS book. Springer, London (1995)
26. Shamir, A.: A survey on mesh segmentation techniques. CGF 27 (September 2008)
27. Shlafman, S., Tal, A., Katz, S.: Metamorphosis of polyhedral surfaces using decomposition. In: CGF, vol. 21, pp. 219–228 (2002)
28. Simari, P., Kalogerakis, E., Singh, K.: Folding meshes: hierarchical mesh segmentation based on planar symmetry. In: Proceedings of the Fourth Eurographics Symposium on Geometry Processing, pp. 111–119. Eurographics Association, Aire-la-Ville (2006)
29. Tierny, J., Vandeborre, J.P., Daoudi, M.: Topology driven 3d mesh hierarchical segmentation. In: Proceedings of the IEEE International Conference on Shape Modeling and Applications 2007, pp. 215–220. IEEE Computer Society, Washington, DC, USA (2007)
30. Várady, T., Kós, G., Benko, P.: Reverse engineering regular objects: Simple segmentation and surface fitting procedures. IJSM (International Journal of Shape Modeling), Procs. of CAGD: New Trends and Applications, Crete 4, 127–142 (1997/1998)
31. Werghi, N., Fisher, R., Ashbrook, A., Robertson, C.: Faithful recovering of quadric surfaces from 3d range data. In: Proceedings 2nd int. Conf. on 3d Digital Imaging and Modeling, pp. 280–289 (1999)
32. Yamauchi, H., Gumhold, S., Zayer, R., Seidel, H.P.: Mesh segmentation driven by gaussian curvature. The Visual Computer 21, 659–668 (2005)
33. Yamauchi, H., Lee, S., Lee, Y., Ohtake, Y., Belyaev, A., Seidel, H.P.: Feature sensitive mesh segmentation with mean shift. In: Proceedings of the International Conference on Shape Modeling and Applications 2005, pp. 238–245. IEEE Computer Society, Washington, DC, USA (2005)
34. Zatzarinni, R., Tal, A., Shamir, A.: Relief analysis and extraction. ACM Trans. Graph. 28(5), 1–9 (2009)
35. Zhang, X., Li, G., Xiong, Y., He, F.: 3D mesh segmentation using mean-shifted curvature. In: Chen, F., Jüttler, B. (eds.) GMP 2008. LNCS, vol. 4975, pp. 465–474. Springer, Heidelberg (2008)
36. Zuckerberger, E., Tal, A., Shlafman, S.: Polyhedral surface decomposition with applications. Computers & Graphics 26(5), 733–743 (2002)

Design of C^2 Spatial Pythagorean-Hodograph Quintic Spline Curves by Control Polygons*

Rida T. Farouki[1], Carla Manni[2], Francesca Pelosi[2], and Maria Lucia Sampoli[3]

[1] Department of Mechanical and Aerospace Engineering, University of California, Davis, CA 95616, USA
[2] Dipartimento di Matematica, Università di Roma Tor Vergata, via della Ricerca Scientifica, 00133 Roma, Italy
[3] Dipartimento di Scienze Matematiche ed Informatiche, Università di Siena, Pian dei Mantellini 44, 53100 Siena, Italy

Abstract. An intuitive approach to designing spatial C^2 Pythagorean–hodograph (PH) quintic spline curves, based on given control polygons, is presented. Although PH curves can always be represented in Bézier or B–spline form, changes to their control polygons will usually compromise their PH nature. To circumvent this problem, an approach similar to that developed in [13] for the planar case is adopted. Namely, the "ordinary" C^2 cubic B–spline curve determined by the given control polygon is first computed, and the C^2 PH spline associated with that control polygon is defined so as to interpolate the nodal points of the cubic B–spline, with analogous end conditions. The construction of spatial PH spline curves is more challenging than the planar case, because of the residual degrees of freedom it entails. Two strategies for fixing these free parameters are presented, based on optimizing shape measures for the PH spline curves.

Keywords: Spatial Pythagorean–hodograph curves; control polygons; cubic B–spline curves; Hermite interpolation; Hopf map; quaternions.

1 Introduction

The construction of free–form curves and surfaces in computer–aided geometric design typically follows one of two paradigms. If precise geometrical constraints must be observed, *interpolation* schemes are most appropriate. If one desires an intuitive means of manipulating the curve/surface shape for aesthetic purposes, on the other hand, the use of a *control point* (e.g., Bézier/B–spline) scheme is recommended. We present an intuitive method for the construction of spatial C^2 Pythagorean–hodograph (PH) quintic spline curves from control polygons.

PH curves [4] incorporate special algebraic structures in their hodographs (derivatives), that offer many useful computational advantages over "ordinary" polynomial parametric curves. For example, their arc lengths are amenable to *exact* computation; they possess *rational* offset curves; and they are well–suited

* This work was partially supported by INdAM Gruppo Nazionale Calcolo Scientifico.

to formulating *real–time CNC interpolator* algorithms, used to drive computer–controlled machines along curved paths with a prescribed speed variation.

Existing methods to construct planar or spatial PH curves are typically based on the interpolation of discrete point/tangent data [5,6,7,8,9,11,12,14]. The design of PH curves using Bézier/B–spline control polygons is more challenging, since they comprise a proper subset of the Bézier/B–spline curves — the control points of PH curves must satisfy certain non–linear constraints. To circumvent this difficulty, a control–polygon approach has been proposed [13] for the case of planar PH splines, in which the control polygon is employed to formulate a "latent" spline interpolation problem. The goal of the present study is to extend this control–polygon design scheme to spatial PH splines. This is non–trivial because the standard (quaternion and Hopf map) representations of spatial PH curves incur one residual degree of freedom per spline segment, and strategies for "optimal" utilization of these residual freedoms must be developed.

A preliminary investigation of the C^2 spatial PH quintic spline interpolation problem was presented in [8]. This employed the quaternion representation, and *ad hoc* algebraic constraints on the coefficients of the quaternion pre–image curve to fix the residual degrees of freedom. The present study differs from [8] in the following key aspects — (1) it is motivated by the desire to formulate an intuitive control–polygon PH spline design scheme, the underlying PH spline interpolation problem remaining hidden from the user; (2) the PH spline interpolation problem is formulated using the Hopf map form, facilitating a simpler implementation in terms of complex coefficients with a clearer geometrical significance; (3) the method exactly reproduces PH cubics when the given data are compatible with such curves; and (4) residual freedoms are used to optimize shape measures of the spline curve associated with the given control polygon, yielding appreciably better curvature and torsion profiles than those produced by the method in [8], as evident in the computed examples presented in Section 5.

In the proposed approach, the C^2 spatial PH quintic spline associated with a given control polygon and knot sequence is defined so as to interpolate the nodal points of the "ordinary" C^2 cubic spline [3] with the same B–spline control points, knot sequence, and end conditions. Further constraints are imposed to fix the additional freedoms associated with spatial PH curves. A key requirement in the proposed scheme is a means of identifying "good" starting approximations — analogous to those used in [6] — to facilitate efficient and robust solution of the optimization problem characterizing the spatial PH quintic spline.

The method permits efficient construction of both open and closed spatial PH splines, that typically agree very closely with the corresponding cubic B–spline curves. The relationship between the PH spline curve and its control polygon is invariant under similarity transformations, and multiple knots may be inserted to reduce the order of continuity to C^1 or C^0 at prescribed points. Also, by using double knots, the PH splines offer a *linear precision* and *local shape modification* capability. Although the non–linear nature of PH splines precludes proofs for certain features of cubic B–splines (e.g., convex–hull confinement) this is of little practical concern in view of the close agreement of the two curves in most cases.

This paper is organized as follows. After briefly reviewing the quaternion and Hopf map representations of spatial PH curves in Section 2, the computation of spatial C^2 PH quintic splines that interpolate given point data under given end conditions is discussed in Section 3, together with the imposition of additional constraints for shape optimization and choice of initial approximations. Section 4 describes how data for the interpolation/optimization scheme is obtained from a given spline control polygon and knot sequence. Finally, Section 5 presents some computed examples, and Section 6 makes some concluding remarks.

2 Spatial PH Quintic Curves

The distinctive feature of a spatial PH curve $\boldsymbol{r}(t) = (x(t), y(t), z(t))$ is that its hodograph $\boldsymbol{r}'(t) = (x'(t), y'(t), z'(t))$ satisfies

$$x'^2(t) + y'^2(t) + z'^2(t) \;=\; \sigma^2(t)$$

for some polynomial $\sigma(t)$. This is equivalent [2] to the requirement that the hodograph $\boldsymbol{r}'(t)$ should be expressible as a quaternion product of the form

$$\boldsymbol{r}'(t) \;=\; \mathcal{A}(t)\,\mathbf{i}\,\mathcal{A}^*(t)\,, \tag{1}$$

where $\mathcal{A}(t) = u(t) + v(t)\,\mathbf{i} + p(t)\,\mathbf{j} + q(t)\,\mathbf{k}$ is a quaternion polynomial of degree m for a PH curve of degree $n = 2m+1$, and $\mathcal{A}^*(t) := u(t) - v(t)\,\mathbf{i} - p(t)\,\mathbf{j} - q(t)\,\mathbf{k}$ is its conjugate. For PH quintics, in particular, we take

$$\mathcal{A}(t) \;:=\; \mathcal{A}_0\,(1-t)^2 + \mathcal{A}_1\,2(1-t)t + \mathcal{A}_2\,t^2\,. \tag{2}$$

An alternative (equivalent) representation is defined in terms of two complex polynomials $\boldsymbol{\alpha}(t)$, $\boldsymbol{\beta}(t)$ by the Hopf map form

$$\boldsymbol{r}'(t) \;=\; (|\boldsymbol{\alpha}(t)|^2 - |\boldsymbol{\beta}(t)|^2, 2\,\mathrm{Re}(\boldsymbol{\alpha}(t)\overline{\boldsymbol{\beta}}(t)), 2\,\mathrm{Im}(\boldsymbol{\alpha}(t)\overline{\boldsymbol{\beta}}(t)))\,. \tag{3}$$

For spatial PH quintics, we use

$$\begin{aligned} \boldsymbol{\alpha}(t) \;&:=\; \boldsymbol{\alpha}_0\,(1-t)^2 + \boldsymbol{\alpha}_1\,2(1-t)t + \boldsymbol{\alpha}_2 t^2\,, \\ \boldsymbol{\beta}(t) \;&:=\; \boldsymbol{\beta}_0\,(1-t)^2 + \boldsymbol{\beta}_1\,2(1-t)t + \boldsymbol{\beta}_2\,t^2\,. \end{aligned} \tag{4}$$

The equivalence of these two forms can be verified by setting

$$\mathcal{A}(t) \;=\; \boldsymbol{\alpha}(t) + \mathbf{k}\,\boldsymbol{\beta}(t)\,, \tag{5}$$

where we identify the quaternion basis element $\mathbf{i}$ with the imaginary unit i. For spatial PH quintics, in particular, we have $\mathcal{A}_r = \boldsymbol{\alpha}_r + \mathbf{k}\,\boldsymbol{\beta}_r$ for $r = 0, 1, 2$.

Once the quaternions $\mathcal{A}_r$ in (2) — or the complex values $\boldsymbol{\alpha}_r, \boldsymbol{\beta}_r$ in (4) — are known, the control points $\boldsymbol{q}_i$ defining the Bézier form

$$\boldsymbol{r}(t) \;=\; \sum_{i=0}^{5} \boldsymbol{q}_i \binom{5}{i} (1-t)^{5-i} t^i$$

of the spatial PH quintic are given [4] by

$$\begin{aligned}
\boldsymbol{q}_1 &= \boldsymbol{q}_0 + \frac{1}{5}\mathcal{A}_0\,\mathbf{i}\,\mathcal{A}_0^*\,, \\
\boldsymbol{q}_2 &= \boldsymbol{q}_1 + \frac{1}{10}(\mathcal{A}_0\,\mathbf{i}\,\mathcal{A}_1^* + \mathcal{A}_1\,\mathbf{i}\,\mathcal{A}_0^*)\,, \\
\boldsymbol{q}_3 &= \boldsymbol{q}_2 + \frac{1}{30}(\mathcal{A}_0\,\mathbf{i}\,\mathcal{A}_2^* + 4\mathcal{A}_1\,\mathbf{i}\,\mathcal{A}_1^* + \mathcal{A}_2\,\mathbf{i}\,\mathcal{A}_0^*)\,, \\
\boldsymbol{q}_4 &= \boldsymbol{q}_3 + \frac{1}{10}(\mathcal{A}_1\,\mathbf{i}\,\mathcal{A}_2^* + \mathcal{A}_2\,\mathbf{i}\,\mathcal{A}_1^*)\,, \\
\boldsymbol{q}_5 &= \boldsymbol{q}_4 + \frac{1}{5}\mathcal{A}_2\,\mathbf{i}\,\mathcal{A}_2^*\,.
\end{aligned} \tag{6}$$

Although the quaternion form has thus far seen more widespread use, we find it more convenient here to work primarily with the Hopf map form since it yields a clearer geometric interpretation of the problem formulation (see Section 3.3). Moreover, the Hopf map form is based on standard complex–number arithmetic, which has been profitably employed [7,13] in the planar case, and significantly simplifies implementation of the numerical methods described below.

3 C^2 Spatial PH Quintic Spline Interpolation

We associate a C^2 spatial PH quintic spline with a specified control polygon and knot sequence by requiring it to interpolate the nodal points of the "ordinary" C^2 cubic spline defined by the prescribed control polygon and knots (see Section 4). Therefore, we must first formulate methods to interpolate point sequences in $\mathbb{R}^3$ corresponding to given parameter values and end conditions. A preliminary study of this problem was presented in [8], but the method employed here incorporates several key improvements that are briefly discussed in the appropriate contexts below. We limit ourselves here to emphasizing that the methods employed below benefit significantly from the results on C^1 Hermite interpolation by spatial PH quintics recently presented in [6].

3.1 C^2 Spatial PH Quintic Spline Equations

Consider the construction of a C^2 spatial PH quintic spline that interpolates a set of points $\boldsymbol{p}_0, \ldots, \boldsymbol{p}_N$ in $\mathbb{R}^3$ for given knots $t_0, \ldots, t_N$. For simplicity, we assume uniform (integer) knots here, but the method can easily be generalized to non–uniform knots. We express the hodograph of the spline segment $\boldsymbol{r}_k(t)$, between $\boldsymbol{p}_{k-1}$ and $\boldsymbol{p}_k$, in the Hopf map form (3) using the quadratic complex polynomials (4), written in terms of complex values $\boldsymbol{u}_{k-1}, \boldsymbol{u}_k, \boldsymbol{u}_{k+1}$ and $\boldsymbol{v}_{k-1}, \boldsymbol{v}_k, \boldsymbol{v}_{k+1}$ as

$$\begin{aligned}
\boldsymbol{\alpha}_k(t) &:= \tfrac{1}{2}(\boldsymbol{u}_{k-1} + \boldsymbol{u}_k)\,(1-t)^2 + \boldsymbol{u}_k\,2(1-t)t + \tfrac{1}{2}(\boldsymbol{u}_k + \boldsymbol{u}_{k+1})\,t^2\,, \\
\boldsymbol{\beta}_k(t) &:= \tfrac{1}{2}(\boldsymbol{v}_{k-1} + \boldsymbol{v}_k)\,(1-t)^2 + \boldsymbol{v}_k\,2(1-t)t + \tfrac{1}{2}(\boldsymbol{v}_k + \boldsymbol{v}_{k+1})\,t^2\,.
\end{aligned} \tag{7}$$

The derivatives of these polynomials are

$$\begin{aligned}\boldsymbol{\alpha}_k'(t) &= (\boldsymbol{u}_k - \boldsymbol{u}_{k-1})\,(1-t) + (\boldsymbol{u}_{k+1} - \boldsymbol{u}_k)\,t\,,\\ \boldsymbol{\beta}_k'(t) &= (\boldsymbol{v}_k - \boldsymbol{v}_{k-1})\,(1-t) + (\boldsymbol{v}_{k+1} - \boldsymbol{v}_k)\,t\,.\end{aligned}$$

Continuity of the first and second derivatives of consecutive segments $\boldsymbol{r}_k(t)$ and $\boldsymbol{r}_{k+1}(t)$ at their juncture $\boldsymbol{r}_k(1) = \boldsymbol{r}_{k+1}(0) = \boldsymbol{p}_k$ is then automatically achieved, since from (3) and (7) we have

$$\boldsymbol{r}_k'(1) \;=\; \boldsymbol{r}_{k+1}'(0) \;=\; (\tfrac{1}{4}(|\boldsymbol{u}_k+\boldsymbol{u}_{k+1}|^2 - |\boldsymbol{v}_k+\boldsymbol{v}_{k+1}|^2), \tfrac{1}{2}(\boldsymbol{u}_k+\boldsymbol{u}_{k+1})(\overline{\boldsymbol{v}}_k+\overline{\boldsymbol{v}}_{k+1}))\,,$$

$$\boldsymbol{r}_k''(1) \;=\; \boldsymbol{r}_{k+1}''(0) \;=\; (|\boldsymbol{u}_{k+1}|^2 - |\boldsymbol{u}_k|^2 + |\boldsymbol{v}_{k+1}|^2 - |\boldsymbol{v}_k|^2, 2(\boldsymbol{u}_{k+1}\overline{\boldsymbol{v}}_{k+1} - \boldsymbol{u}_k\overline{\boldsymbol{v}}_k))\,.$$

It remains to ensure that each segment $\boldsymbol{r}_k(t)$ satisfies the end–point interpolation conditions $\boldsymbol{r}_k(0) = \boldsymbol{p}_{k-1}$ and $\boldsymbol{r}_k(1) = \boldsymbol{p}_k$, i.e., that

$$\int_0^1 \boldsymbol{r}_k'(t)\,\mathrm{d}t \;=\; \Delta\boldsymbol{p}_k \;:=\; \boldsymbol{p}_k - \boldsymbol{p}_{k-1}\,,$$

or, setting $\Delta\boldsymbol{p}_k =: (\Delta x_k, \Delta y_k, \Delta z_k)$ and using (3),

$$\int_0^1 |\boldsymbol{\alpha}_k(t)|^2 - |\boldsymbol{\beta}_k(t)|^2\,\mathrm{d}t \;=\; \Delta x_k\,, \quad \int_0^1 2\,\boldsymbol{\alpha}_k(t)\overline{\boldsymbol{\beta}}_k(t)\,\mathrm{d}t \;=\; \Delta y_k + \mathrm{i}\,\Delta z_k\,.$$

Substituting (7) and evaluating the integrals, we obtain

$$\begin{aligned}&|\boldsymbol{u}_{k-1}+\boldsymbol{u}_k|^2 - |\boldsymbol{v}_{k-1}+\boldsymbol{v}_k|^2\\ &+ (\boldsymbol{u}_{k-1}+\boldsymbol{u}_k)\overline{\boldsymbol{u}}_k + (\overline{\boldsymbol{u}}_{k-1}+\overline{\boldsymbol{u}}_k)\boldsymbol{u}_k - (\boldsymbol{v}_{k-1}+\boldsymbol{v}_k)\overline{\boldsymbol{v}}_k - (\overline{\boldsymbol{v}}_{k-1}+\overline{\boldsymbol{v}}_k)\boldsymbol{v}_k\\ &+ \tfrac{1}{6}(16|\boldsymbol{u}_k|^2 + (\boldsymbol{u}_{k-1}+\boldsymbol{u}_k)(\overline{\boldsymbol{u}}_k+\overline{\boldsymbol{u}}_{k+1}) + (\overline{\boldsymbol{u}}_{k-1}+\overline{\boldsymbol{u}}_k)(\boldsymbol{u}_k+\boldsymbol{u}_{k+1}))\\ &- \tfrac{1}{6}(16|\boldsymbol{v}_k|^2 + (\boldsymbol{v}_{k-1}+\boldsymbol{v}_k)(\overline{\boldsymbol{v}}_k+\overline{\boldsymbol{v}}_{k+1}) + (\overline{\boldsymbol{v}}_{k-1}+\overline{\boldsymbol{v}}_k)(\boldsymbol{v}_k+\boldsymbol{v}_{k+1}))\\ &+ \boldsymbol{u}_k(\overline{\boldsymbol{u}}_k+\overline{\boldsymbol{u}}_{k+1}) + \overline{\boldsymbol{u}}_k(\boldsymbol{u}_k+\boldsymbol{u}_{k+1}) - \boldsymbol{v}_k(\overline{\boldsymbol{v}}_k+\overline{\boldsymbol{v}}_{k+1}) - \overline{\boldsymbol{v}}_k(\boldsymbol{v}_k+\boldsymbol{v}_{k+1})\\ &+ |\boldsymbol{u}_k+\boldsymbol{u}_{k+1}|^2 - |\boldsymbol{v}_k+\boldsymbol{v}_{k+1}|^2 \;=\; 20\,\Delta x_k\,,\end{aligned}$$

$$\begin{aligned}&(\boldsymbol{u}_{k-1}+\boldsymbol{u}_k)(\overline{\boldsymbol{v}}_{k-1}+\overline{\boldsymbol{v}}_k)\\ &+ (\boldsymbol{u}_{k-1}+\boldsymbol{u}_k)\overline{\boldsymbol{v}}_k + \boldsymbol{u}_k(\overline{\boldsymbol{v}}_{k-1}+\overline{\boldsymbol{v}}_k)\\ &+ \tfrac{1}{6}((\boldsymbol{u}_{k-1}+\boldsymbol{u}_k)(\overline{\boldsymbol{v}}_k+\overline{\boldsymbol{v}}_{k+1}) + 16\boldsymbol{u}_k\overline{\boldsymbol{v}}_k + (\boldsymbol{u}_k+\boldsymbol{u}_{k+1})(\overline{\boldsymbol{v}}_{k-1}+\overline{\boldsymbol{v}}_k))\\ &+ \boldsymbol{u}_k(\overline{\boldsymbol{v}}_k+\overline{\boldsymbol{v}}_{k+1}) + (\boldsymbol{u}_k+\boldsymbol{u}_{k+1})\overline{\boldsymbol{v}}_k\\ &+ (\boldsymbol{u}_k+\boldsymbol{u}_{k+1})(\overline{\boldsymbol{v}}_k+\overline{\boldsymbol{v}}_{k+1}) \;=\; 10(\Delta y_k + \mathrm{i}\,\Delta z_k)\,.\end{aligned}$$

These equations can be reduced to

$$\begin{aligned}&6\,|\boldsymbol{u}_{k-1}|^2 + 54\,|\boldsymbol{u}_k|^2 + 6\,|\boldsymbol{u}_{k+1}|^2 - 6\,|\boldsymbol{v}_{k-1}|^2 - 54\,|\boldsymbol{v}_k|^2 - 6\,|\boldsymbol{v}_{k+1}|^2\\ &+ 13\,(\boldsymbol{u}_{k-1}\overline{\boldsymbol{u}}_k + \overline{\boldsymbol{u}}_{k-1}\boldsymbol{u}_k + \boldsymbol{u}_k\overline{\boldsymbol{u}}_{k+1} + \overline{\boldsymbol{u}}_k\boldsymbol{u}_{k+1})\\ &- 13\,(\boldsymbol{v}_{k-1}\overline{\boldsymbol{v}}_k + \overline{\boldsymbol{v}}_{k-1}\boldsymbol{v}_k + \boldsymbol{v}_k\overline{\boldsymbol{v}}_{k+1} + \overline{\boldsymbol{v}}_k\boldsymbol{v}_{k+1})\end{aligned}$$

$$+\,\boldsymbol{u}_{k-1}\overline{\boldsymbol{u}}_{k+1}+\overline{\boldsymbol{u}}_{k-1}\boldsymbol{u}_{k+1}-\boldsymbol{v}_{k-1}\overline{\boldsymbol{v}}_{k+1}-\overline{\boldsymbol{v}}_{k-1}\boldsymbol{v}_{k+1} \;=\; 120\,\Delta x_k\,, \tag{8}$$

$$\begin{aligned}&6\,\boldsymbol{u}_{k-1}\overline{\boldsymbol{v}}_{k-1}+54\,\boldsymbol{u}_k\overline{\boldsymbol{v}}_k+6\,\boldsymbol{u}_{k+1}\overline{\boldsymbol{v}}_{k+1}\\&+13\,(\boldsymbol{u}_{k-1}+\boldsymbol{u}_{k+1})\overline{\boldsymbol{v}}_k+13\,\boldsymbol{u}_k(\overline{\boldsymbol{v}}_{k-1}+\overline{\boldsymbol{v}}_{k+1})\\&+\boldsymbol{u}_{k-1}\overline{\boldsymbol{v}}_{k+1}+\boldsymbol{u}_{k+1}\overline{\boldsymbol{v}}_{k-1} \;=\; 60\,(\Delta y_k+\mathrm{i}\,\Delta z_k)\,.\end{aligned} \tag{9}$$

Conditions (8) and (9) for $k=1,\ldots,N$ specify N real and N complex equations in the $2N$ complex variables $\boldsymbol{u}_1,\ldots,\boldsymbol{u}_N$ and $\boldsymbol{v}_1,\ldots,\boldsymbol{v}_N$.

3.2 End Conditions

End conditions must also be imposed on the system (8)–(9), in order to account for the undefined quantities $\boldsymbol{u}_0,\boldsymbol{v}_0$ and $\boldsymbol{u}_{N+1},\boldsymbol{v}_{N+1}$ in the cases $k=1,N$. For open curves, *cubic end spans* are appropriate. They correspond to the case where the $k=1,N$ instances of (7) are degree–elevated linear polynomials, defined by the conditions

$$\begin{aligned}(\boldsymbol{u}_0,\boldsymbol{v}_0) &= (2\boldsymbol{u}_1-\boldsymbol{u}_2,2\boldsymbol{v}_1-\boldsymbol{v}_2)\,,\\(\boldsymbol{u}_{N+1},\boldsymbol{v}_{N+1}) &= (2\boldsymbol{u}_N-\boldsymbol{u}_{N-1},2\boldsymbol{v}_N-\boldsymbol{v}_{N-1})\,,\end{aligned} \tag{10}$$

rather than true quadratics. Alternatively, end derivatives can be imposed, i.e., for given vectors $\boldsymbol{d}_0$, $\boldsymbol{d}_N$ we set

$$\begin{aligned}(\boldsymbol{u}_0,\boldsymbol{v}_0) &= (2\tilde{\boldsymbol{u}}_0-\boldsymbol{u}_1,2\tilde{\boldsymbol{v}}_0-\boldsymbol{v}_1)\,,\\(\boldsymbol{u}_{N+1},\boldsymbol{v}_{N+1}) &= (2\tilde{\boldsymbol{u}}_N-\boldsymbol{u}_N,2\tilde{\boldsymbol{v}}_N-\boldsymbol{v}_N)\,,\end{aligned} \tag{11}$$

where $\tilde{\boldsymbol{u}}_0,\tilde{\boldsymbol{v}}_0$ and $\tilde{\boldsymbol{u}}_{N+1},\tilde{\boldsymbol{v}}_{N+1}$ are complex numbers such that[1]

$$\boldsymbol{r}_0'(0) \;=\; \boldsymbol{d}_0\,, \qquad \boldsymbol{r}_N'(1) \;=\; \boldsymbol{d}_N\,. \tag{12}$$

For smooth closed C^2 curves, *periodic end conditions* must be used: $\boldsymbol{u}_0,\ldots,\boldsymbol{u}_{N+1}$ and $\boldsymbol{v}_0,\ldots,\boldsymbol{v}_{N+1}$ are regarded as cyclical lists, with

$$(\boldsymbol{u}_0,\boldsymbol{v}_0) \;=\; (\boldsymbol{u}_N,\boldsymbol{v}_N)\,, \qquad (\boldsymbol{u}_{N+1},\boldsymbol{v}_{N+1}) \;=\; (\boldsymbol{u}_1,\boldsymbol{v}_1)\,, \tag{13}$$

in order to ensure that

$$\boldsymbol{r}_N'(1) \;=\; \boldsymbol{r}_1'(0)\,, \qquad \boldsymbol{r}_N''(1) \;=\; \boldsymbol{r}_1''(0)\,.$$

[1] Since each of these conditions imposes three scalar constraints on four variables, there is a one–parameter family of possible values for $\tilde{\boldsymbol{u}}_0,\tilde{\boldsymbol{v}}_0$ and $\tilde{\boldsymbol{u}}_{N+1},\tilde{\boldsymbol{v}}_{N+1}$. Invoking the Hermite interpolation procedures presented in [6], a simple and efficient selection of $\tilde{\boldsymbol{u}}_0,\tilde{\boldsymbol{v}}_0$ and $\tilde{\boldsymbol{u}}_{N+1},\tilde{\boldsymbol{v}}_{N+1}$ can be achieved (see Section 3.4) in a manner that reproduces PH cubics. Note that the case of specified end derivatives was not treated in [8].

3.3 Additional Constraints

Equations (8)–(9), amended by the end conditions, specify $3N$ scalar conditions on the $4N$ degrees of freedom embodied in the $2N$ complex unknowns $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N$. Hence, there are N residual degrees of freedom, and a means of fixing them must be specified to uniquely define the PH spline curve.

A preliminary study of interpolation by spatial C^2 PH quintic splines was presented in [8], using Newton–Raphson–like iterations based on the quaternion representation. The residual freedoms were fixed by imposing a purely algebraic condition (without an intuitive geometric meaning) on the quaternion coefficients — see Section 4 of [8]. In the present context, we wish to ensure that the method will reproduce a cubic PH curve, whenever the given data are compatible with such a curve (a property that the method in [8] does not possess).

Our preference here is thus to adopt additional constraints that have a clearer geometrical interpretation, based on the Hopf map representation. Two possible approaches are proposed to determine a C^2 PH spline curve that interpolates the data $\boldsymbol{p}_0, \ldots, \boldsymbol{p}_N$. The first involves minimizing a global shape measure for the interpolant, subject to the interpolation constraints, while the second entails imposing one additional scalar constraint per spline segment.

The first approach seeks to minimize the “distance” of the PH quintic spline from a (single) PH cubic. With this approach, when the data points are sampled from a PH cubic, the method exactly reproduces that PH cubic. Subject to the interpolation constraints (8)–(9) with appropriate end conditions, the expression minimized in this approach is

$$F_1(\boldsymbol{u}_1, \boldsymbol{v}_1, \ldots, \boldsymbol{u}_N, \boldsymbol{v}_N) := \sum_{k=2}^{N-1} |\boldsymbol{u}_{k-1} - 2\boldsymbol{u}_k + \boldsymbol{u}_{k+1}|^2 + |\boldsymbol{v}_{k-1} - 2\boldsymbol{v}_k + \boldsymbol{v}_{k+1}|^2 \,. \tag{14}$$

This form is motivated by the observation that, when

$$\boldsymbol{u}_k = \tfrac{1}{2}(\boldsymbol{u}_{k-1} + \boldsymbol{u}_{k+1}) \qquad \text{and} \qquad \boldsymbol{v}_k = \tfrac{1}{2}(\boldsymbol{v}_{k-1} + \boldsymbol{v}_{k+1}) \,, \tag{15}$$

expressions (7) define degree–elevated linear polynomials rather than quadratics, so each segment $\boldsymbol{r}_k(t)$ is a degree–elevated PH cubic rather than a PH quintic.

The second approach employs one additional (scalar) constraint per segment, based on the distance between successive pairs of complex values $(\boldsymbol{u}_k, \boldsymbol{v}_k)$ in $\mathbb{C}^2$. Namely, we require that

$$|\boldsymbol{u}_k - \boldsymbol{u}_{k-1}|^2 + |\boldsymbol{v}_k - \boldsymbol{v}_{k-1}|^2 = |\boldsymbol{u}_{k+1} - \boldsymbol{u}_k|^2 + |\boldsymbol{v}_{k+1} - \boldsymbol{v}_k|^2 \tag{16}$$

for $k = 1, \ldots, N$. In conjunction with equations (8)–(9), under appropriate end conditions, the constraints (16) define a system of $4N$ quadratic equations in $4N$ real unknowns. One can easily verify that, when conditions (15) hold, conditions (16) are satisfied. Hence, this approach also reproduces PH cubics.

It should be noted that, even when additional constraints are introduced to fix the residual degrees of freedom, the nonlinear structure of equations (8), (9), and

(16) usually implies a lack of solution uniqueness. Further criteria to choose the "best" interpolant among all possible solutions are usually necessary [8,13]. Note also that equations (8)–(9) and (16) depend on the complex variables $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N$ *and* their conjugates. Since the conjugate of a complex variable is not an analytic function of that variable, numerical methods that make use of derivatives (such as the Newton–Raphson iteration) are not directly applicable.[2] Instead of solving the system using a Newton–like method, we prefer to determine the C^2 PH quintic spline interpolant to given data by a suitable minimization procedure in the second approach, as well as the first. Actually, we minimize the constant function

$$F_2(\boldsymbol{u}_1, \boldsymbol{v}_1, \ldots, \boldsymbol{u}_N, \boldsymbol{v}_N) \, := 1 \,, \tag{17}$$

under the constraints (8)–(9), under suitable boundary conditions, and (16).

3.4 Starting Approximation

For both methods, a good starting approximation is critical for convergence to the "good" PH spline interpolant. To obtain such a starting approximation, the "ordinary" C^2 cubic spline interpolating the points $\boldsymbol{p}_0, \ldots, \boldsymbol{p}_N$ at uniform knots, under appropriate end conditions, is used as a reference curve.

In the planar case, the starting approximation is obtained by equating mid-point derivatives for each ordinary cubic and PH quintic spline segment. These conditions incur a quadratic system, but it is possible to obtain the solution by solving just a linear system [7,13]. In the spatial case, however, it is not possible to compute the starting approximation by an analogous approach because of the residual degrees of freedom associated with the quaternion and Hopf map forms. Specifically, equating mid–point derivatives yields equations of the form

$$\mathcal{A} \, \mathbf{i} \, \mathcal{A}^* \, = \, \boldsymbol{d} \,,$$

for a given vector $\boldsymbol{d}$, which admit [5] a one–parameter family of solutions for the quaternion $\mathcal{A}$. In view of this, we adopt a different strategy. First, we compute C^1 PH quintic Hermite segments interpolating the data points and derivatives of the cubic spline. Each PH segment is constructed using the CC strategy described in [6]. This procedure is easy to implement, and reproduces PH cubics when the data are compatible with them. Now if $\boldsymbol{r}_k'(t) = \mathcal{A}_k(t) \, \mathbf{i} \, \mathcal{A}_k^*(t)$ is the hodograph of the spline segment $\boldsymbol{r}_k(t)$, where

$$\mathcal{A}_k(t) \, = \, \mathcal{A}_{k,0}(1-t)^2 \, + \, \mathcal{A}_{k,1} 2(1-t)t \, + \, \mathcal{A}_{k,2} t^2 \,,$$

it remains unchanged upon replacing $\mathcal{A}_{k,j}$ by $\mathcal{A}_{k,j} \mathcal{R}_k$ for $j = 0, 1, 2$ where

$$\mathcal{R}_k \, = \, \cos \phi_k + \mathbf{i} \, \sin \phi_k \,.$$

Thus, following [8], for open curves we determine suitable angles ϕ_k so that

$$\mathcal{A}_{k+1,0} \mathcal{R}_{k+1} \, = \, \mathcal{A}_{k,2} \mathcal{R}_k \qquad \text{for} \;\; k = 1, \ldots, N-1 \,.$$

[2] An alternative is to write (8)–(9) and (16) explicitly as real equations, in terms of the real and imaginary parts of $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N$.

A similar strategy is used for closed curves, see [8] for details. After selecting the angles ϕ_k, in accordance with (5) we set

$$\mathcal{A}_{k,1}\mathcal{R}_k \ =: \ \boldsymbol{u}_k + \mathbf{k}\boldsymbol{v}_k \qquad \text{for} \ \ k = 1, \ldots, N$$

as starting values of $(\boldsymbol{u}_k, \boldsymbol{v}_k)$. In the case of specified end derivatives for open curves, the values $\tilde{\boldsymbol{u}}_0, \tilde{\boldsymbol{v}}_0$ and $\tilde{\boldsymbol{u}}_N, \tilde{\boldsymbol{v}}_N$ in (11) are fixed by setting

$$\mathcal{A}_{1,0}\mathcal{R}_1 \ =: \ \tilde{\boldsymbol{u}}_0 + \mathbf{k}\tilde{\boldsymbol{v}}_0 \,, \qquad \mathcal{A}_{N,2}\mathcal{R}_N \ =: \ \tilde{\boldsymbol{u}}_N + \mathbf{k}\tilde{\boldsymbol{v}}_N \,. \tag{18}$$

4 Control Polygons for C^2 PH Quintic Splines

Based on the interpolation scheme described above, we can formulate a control–polygon approach to the design of C^2 spatial PH quintic spline curves that closely parallels the methods used [13] for the planar case. For completeness, we briefly summarize the means of obtaining the data for the interpolation problem.

For open curves, let

$$t_0 = t_1 = t_2 = t_3 < \cdots < t_{M+1} = t_{M+2} = t_{M+3} = t_{M+4}, \tag{19}$$

with $t_{3+k} := k$ for $k = 0, \ldots, M-2$ be a uniform knot sequence with four–fold end knots, and let the control points

$$\boldsymbol{P}_0, \ldots, \boldsymbol{P}_M \tag{20}$$

define the associated control polygon. Consider the C^2 cubic B-spline curve $\boldsymbol{c}(t)$ specified [1] by the given control points and knots, with nodal points

$$\boldsymbol{p}_k \ := \ \boldsymbol{c}(t_{3+k}) \qquad \text{for} \quad k = 0, \ldots, M-2 \tag{21}$$

and end derivatives

$$\boldsymbol{d}_0 \ := \ \boldsymbol{c}'(t_3) \,, \qquad \boldsymbol{d}_N \ := \ \boldsymbol{c}'(t_{M+1}) \,. \tag{22}$$

The open C^2 PH quintic spline curve associated with the knots (19) and control points (20) is then defined as the interpolant to the $M-1 =: N+1$ nodal points (21), with end derivatives defined by (11), (18), and (22).

A similar approach can be used to construct closed PH splines by specifying periodic knot sequences, control polygons, and invoking the end conditions (13). Multiple knots can be inserted, to reduce the order of continuity in a controlled manner (from C^2 to C^1 or C^0) at the nodal parameter values, exactly as in the planar case — see [13] for details, and a discussion of the linear precision and local shape modification properties of the PH quintic splines.

For simplicity, our focus here has been on the case of uniform knots, but the generalization to non–uniform knots is not difficult (see [8] for the formulation using the quaternion representation). However, as noted in [13], the nodal points of a cubic B–spline curve tend to be more uniformly distributed when the control points are unevenly spaced, since they are weighted averages of the latter. Thus, shape improvements obtained with non–uniform knots, as compared to uniform knots, are rather modest for curves with reasonable control polygons.

5 Computed Examples

The examples presented here were computed through pilot implementations of the two methods in `MATLAB`. General–purpose implementations in a high–level programming language will require more detailed attention to the efficiency of the numerical solution procedure, in order to ensure interactive control–polygon manipulation of the spatial C^2 PH quintic spline curves.

Although the cost functions (14) and (17) are just quadratic and constant, respectively, the optimization problems require special attention to the specified constraints. A common efficient paradigm for solving this kind of problem is to transform it into an easier sub–problem that can be readily solved, and used as the basis of an iterative process. The `MATLAB` implementation employed here uses an iterative quadratic programming scheme, in which the quadratic sub–problem at each iteration is solved through an active set strategy [10].

The method is very efficient and requires, for the examples presented below, ~ 2 seconds cpu time on a typical lap–top computer when minimization of F_1 is used, minimization of F_2 is about five times faster[3]. Unless otherwise stated, the first approach (i.e., minimization of F_1) is used in the following examples.

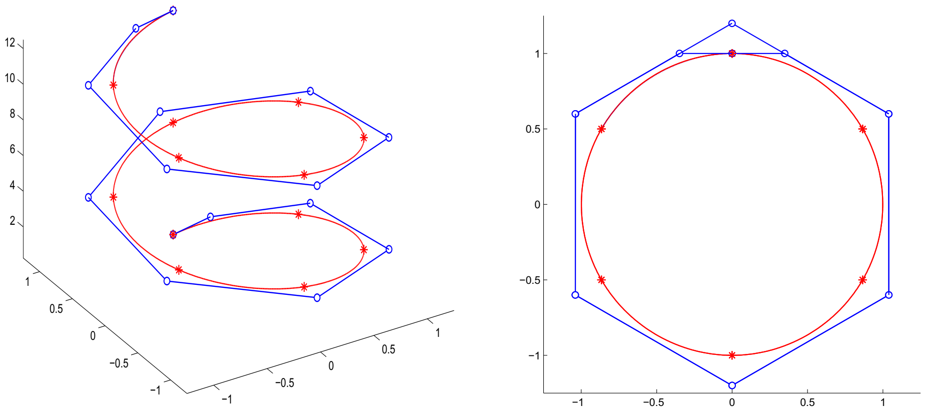


Fig. 1. Example 1. Left: control polygon and corresponding C^2 PH quintic spline curve (solid) and C^2 cubic B–spline (dotted) for point data sampled from a circular helix — the two curves are virtually indistinguishable. Right: projection of the control polygon and the two curves onto the (x, y) plane.

[3] The method in [8] is about ten times faster but, as is evident in the examples below, it produces curves of lower quality.

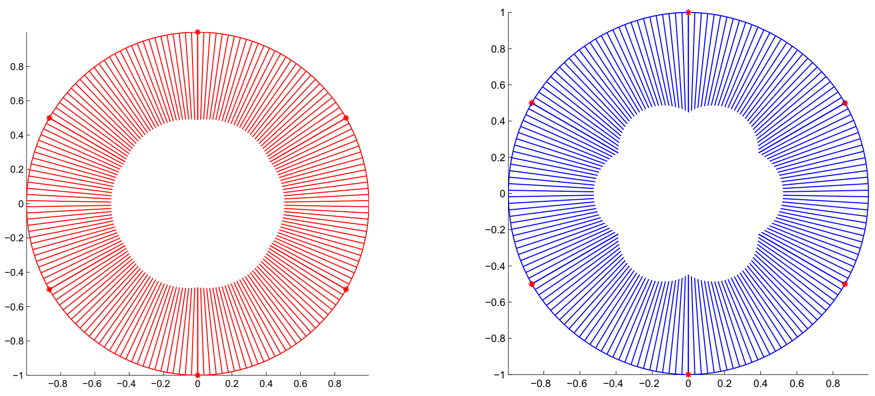


Fig. 2. Example 1. Planar projections of the curvature "porcupine" plots for the curves in Example 1. Left: the PH quintic spline curve. Right: the cubic B–spline curve.

Example 1. In this example, we use the control polygon for a C^2 cubic B–spline curve that interpolates points uniformly sampled from a circular helix, as follows:

$$\boldsymbol{p}_i = (\sin(t_i), \cos(t_i), t_i)\,, \quad t_i = \frac{i}{N}\,4\pi, \quad i = 0, \ldots, N,\ N = 12.$$

Fig. 1 shows the resulting C^2 PH quintic spline, together with the control polygon and corresponding cubic B–spline. The two curves appear almost identical. However, a closer inspection (Figs. 2–4) shows that the resulting PH spline exhibits better curvature and torsion profiles than the "ordinary" cubic B–spline. Minimization of F_2 rather than F_1 gives somewhat improved results,[4] as evident from the torsion profiles in Fig. 4. In Fig. 5 we compare curvature and torsion profiles for the PH splines generated by the present method and by the method of [8], which indicate that the former produces a much better approximation of the helix.

Including a double knot in the knot sequence results in a point of tangency of the curve with the control polygon, the order of continuity at this point being reduced to C^1. A close–up shows that the PH spline still remains very close to the cubic B–spline — see Fig. 6. In Fig. 7 we compare a PH quintic spline and cubic B–spline, with one double knot and one triple knot — a comparison of the curvature and torsion profiles is shown in Fig. 8.

Example 2. The second example involves a C^2 closed curve interpolating:

$$\begin{array}{lll}\boldsymbol{p}_0 = (5,-1,\frac{5}{2}), & \boldsymbol{p}_1 = (5,1,\frac{5}{2}), & \boldsymbol{p}_2 = (2,\frac{3}{2},\frac{2}{5}),\\ \boldsymbol{p}_3 = (-2,\frac{3}{2},1), & \boldsymbol{p}_4 = (-5,1,\frac{5}{2}), & \boldsymbol{p}_5 = (-5,-1,\frac{5}{2}),\\ \boldsymbol{p}_6 = (-2,-\frac{3}{2},\frac{2}{5}), & \boldsymbol{p}_7 = (2,-\frac{3}{2},1), & \boldsymbol{p}_8 = (5,-1,\frac{5}{2});\end{array}$$

under periodic end conditions, as shown in Fig. 9. Figs. 10 and 11 show the curvature and torsion plots obtained with the two approaches for this case.

[4] For an exact circular helix, the curvature and torsion are both precisely constant.

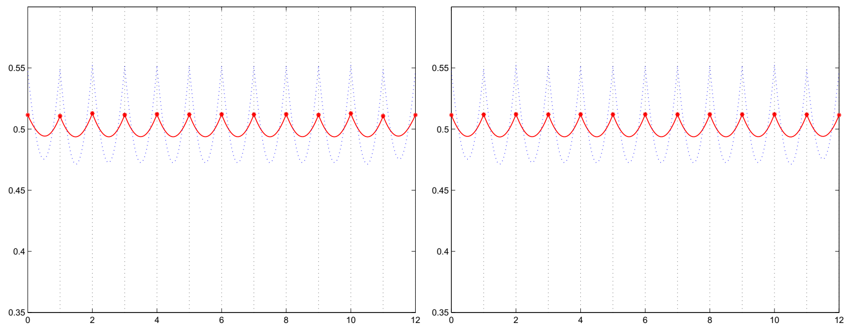


Fig. 3. Example 1. Curvature profiles for the PH quintic spline curve (solid) and cubic B–spline curve (dotted). Left: the first method, based on minimizing (14). Right: the second method, based on minimizing (17) under the constraints (16).

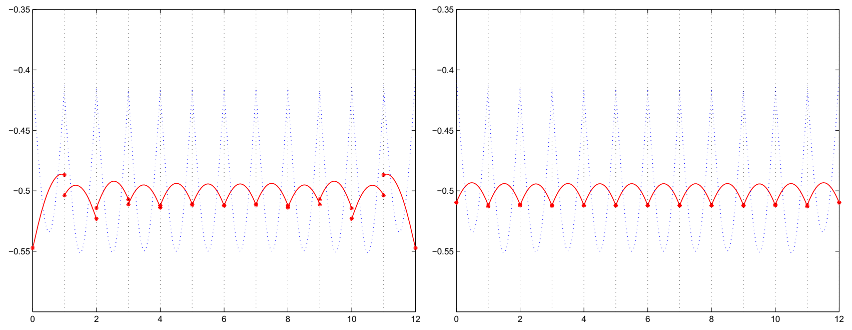


Fig. 4. Example 1. Torsion profiles for the PH quintic spline curve (solid) and cubic B–spline curve (dotted). Left: the first method, based on minimizing (14). Right: the second method, based on minimizing (17) under the constraints (16).

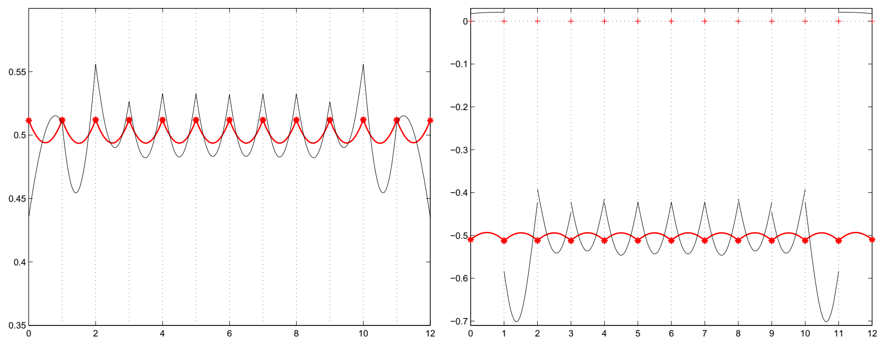


Fig. 5. Example 1. Comparison of curvature (left) and torsion (right) profiles obtained (thin lines) by the interpolation method described in [8], and (thick lines) by the method based on minimization of (17) under the constraints (16). The substantial improvement afforded by the latter method is evident in both the curvature and torsion. In particular, the poor behavior at both ends incurred by the end conditions used in [8] — as already noted in Section 6 of [8] — is completely absent with the new approach.

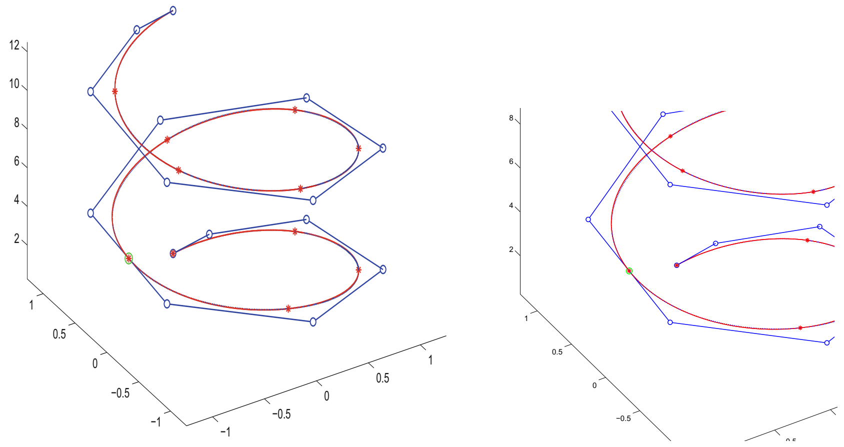


Fig. 6. Example 1. PH quintic spline curve (solid) and cubic B–spline curve (dotted) defined by the knot sequence $\{0, 0, 0, 0, 1, 2, 3, 4, 4, 5, 6, 7, 8, 9, 10, 11, 11, 11, 11\}$ with a double knot. Right: a close–up, showing the point of tangency to the control polygon.

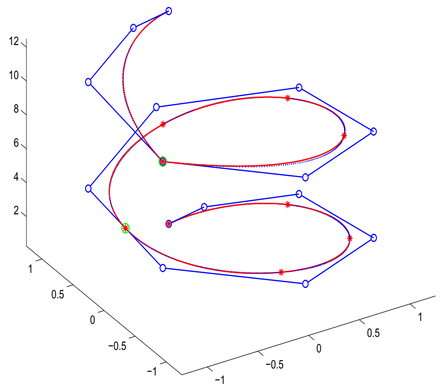


Fig. 7. Example 1. PH quintic spline (solid) and cubic B–spline (dotted) for the knot vector $\{0, 0, 0, 0, 1, 2, 3, 4, 4, 5, 6, 7, 8, 8, 8, 9, 9, 9, 9\}$ with interior double and triple knots.

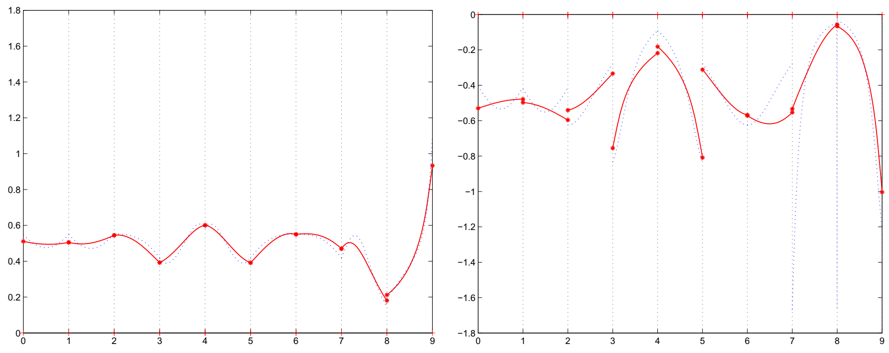


Fig. 8. Example 1. Comparison of the curvature (left) and torsion (right) profiles for the PH quintic spline curve and cubic B–spline curve shown in Fig. 7.

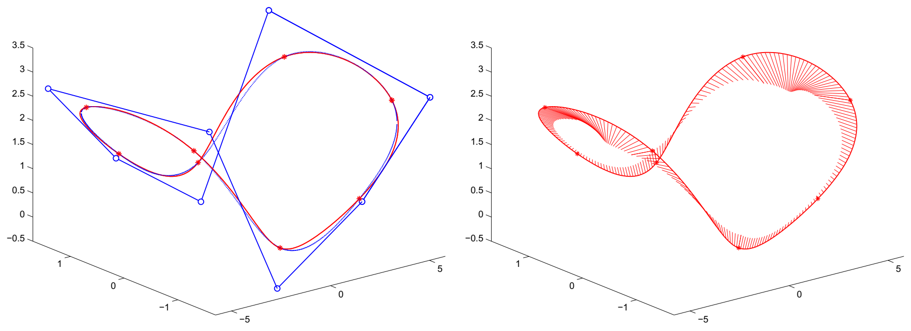


Fig. 9. Example 2. Control polygon and corresponding PH quintic spline (solid) and cubic B–spline (dotted), for periodic end conditions. Right: curvature "porcupine" plot.

The first approach, based on minimizing (14), gives somewhat better results (as seen in the torsion plot). Fig. 12 compares curvature and torsion profiles for the present method and the approach presented in [8]. Fig. 13 compares a PH quintic spline and cubic B–spline for a knot sequence with one double and one triple knot, and a comparison of the curvature and torsion profiles is shown in Fig. 14.

Example 3. The third example compares the performance of the present method with the method previously described in [8], in particular with respect to the exact reproduction of a PH cubic curve when the given data is compatible with such a curve. We consider the cubic curve interpolating the Hermite data

$$\boldsymbol{p}_0 = (0,0,0), \quad \boldsymbol{d}_0 = (1,0,0), \quad \boldsymbol{p}_1 = (\tfrac{33}{100}, \tfrac{7}{10}, \tfrac{4}{15}), \quad \boldsymbol{d}_1 = (-\tfrac{1}{100}, 2, -\tfrac{1}{5})$$

on $t \in [\,0, 1\,]$. This cubic is actually a PH curve, and we sample it at three and four evenly spaced points (see Fig. 15). The present method exactly reproduces the PH cubic, while the method of [8] fails to do so. This is particularly unfortunate in the context of the present paper, i.e., designing PH curves based on control polygons. Indeed the shape of the PH quintic spline obtained by the method of [8] does not correlate well with the associated control polygon.

6 Closure

Methods for the design of spatial C^2 PH quintic spline curves by specification of control polygons and corresponding knot sequences have been presented. The approach is based on interpolating the nodal points of the "ordinary" C^2 cubic B–spline curve defined by the given control points and knots. The distinctive feature of the spatial PH curve design scheme, as compared to planar PH curve constructions, is the need to account for the residual degrees of freedoms in the interpolation problem, so as to obtain curves with desirable shape properties.

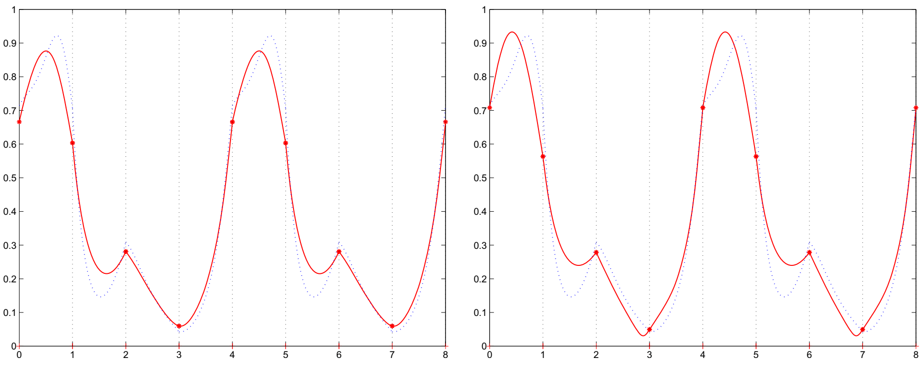


Fig. 10. Example 2. Curvature plots for the PH quintic spline (solid) and cubic B–spline (dotted) in Fig. 9. Left: based on minimizing (14). Right: using conditions (16).

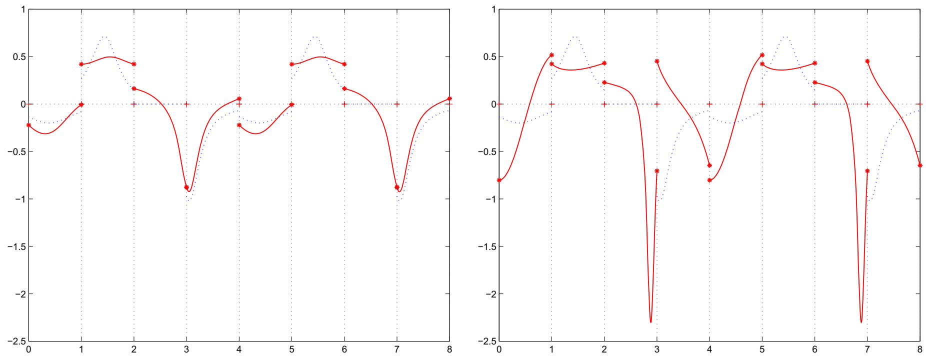


Fig. 11. Example 2. Torsion plots for the PH quintic spline (solid) and cubic B–spline (dotted) in Fig. 9. Left: based on minimization of (14). Right: based on conditions (16).

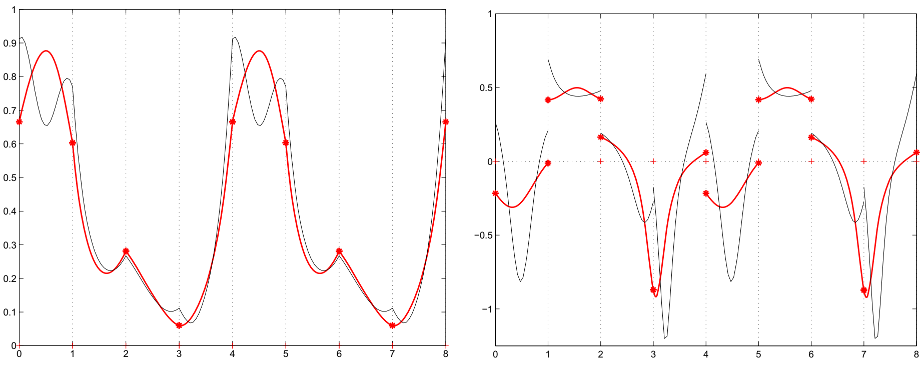


Fig. 12. Example 2. Comparison of curvature (left) and torsion (right) obtained (thin lines) by the method described in [8] and (thick lines) the method based on minimizing (14). The present method yields a significant improvement over that of [8], by producing smoother variations of the curvature and torsion.

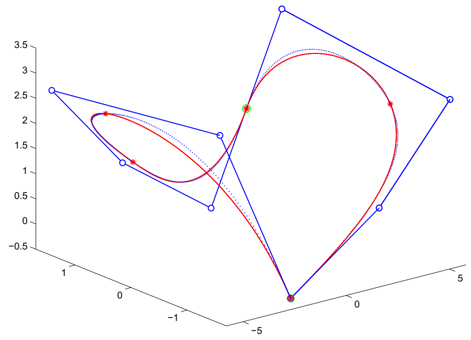


Fig. 13. Example 2. Cubic B–spline (dotted) and PH quintic spline (solid) for a knot sequence $\{-1,-1,-1,0,1,1,2,3,4,4,4,5,6,6,7\}$ with interior double and triple knots.

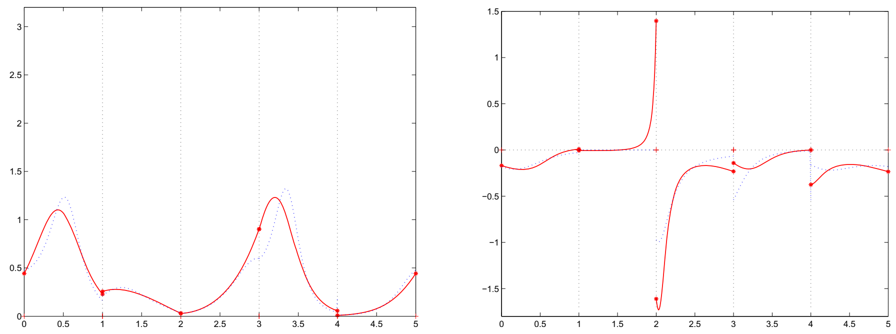


Fig. 14. Example 2. Comparison of curvature (left) and torsion (right) profiles for the PH quintic and cubic B–spline curves with double and triple knots, shown in Fig. 13.

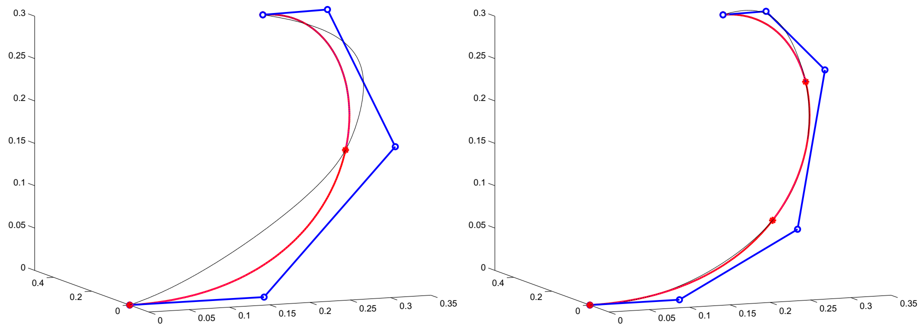


Fig. 15. Example 3. Comparison of PH quintic spline produced (thick lines) by the present method and (thin lines) by the method in [8]. The present method exactly reproduces the PH cubic. The PH quintic splines obtained (for different samplings) by the earlier method exhibit odd behavior, that does not agree well with the control polygon (the discrepancy diminishes upon increasing the number of sample points).

Two alternative methods to fix these residual freedoms were proposed herein, that both reproduce PH cubics when the given data is compatible with a single PH cubic. The computed examples show that they generate spatial PH quintic spline curves of shape quality comparable to, or better than, the corresponding "ordinary" cubic B–spline curves. The underlying PH quintic spline interpolation scheme incorporates several improvements over the method previously described in [8], including use of only complex–number arithmetic; exact reproduction of PH cubics; exploiting the residual freedoms for shape optimization; and refined starting approximations. The present study is preliminary in nature, and further research on enhancing the efficiency of the proposed schemes may be needed to provide practical interactive design tools.

Acknowledgements. The authors thank the referees for their comments and suggestions.

References

1. de Boor, C.: A Practical Guide to Splines. Springer, New York (2001)
2. Choi, H.I., Lee, D.S., Moon, H.P.: Clifford algebra, spin representation, and rational parameterization of curves and surfaces. Adv. Comp. Math. 17, 5–48 (2002)
3. Farin, G.: Curves and Surfaces for Computer Aided Geometric Design, 4th edn. Academic Press, San Diego (1997)
4. Farouki, R.T.: Pythagorean–Hodograph Curves: Algebra and Geometry Inseparable. Springer, Berlin (2008)
5. Farouki, R.T., al–Kandari, M., Sakkalis, T.: Hermite interpolation by rotation–invariant spatial Pythagorean–hodograph curves. Adv. Comp. Math. 17, 369–383 (2002)
6. Farouki, R.T., Giannelli, C., Manni, C., Sestini, A.: Identification of spatial PH quintic Hermite interpolants with near–optimal shape measures. Comput. Aided Geom. Design 25, 274–297 (2008)
7. Farouki, R.T., Kuspa, B.K., Manni, C., Sestini, A.: Efficient solution of the complex quadratic tridiagonal system for C^2 PH quintic splines. Numer. Algor. 27, 35–60 (2001)
8. Farouki, R.T., Manni, C., Sestini, A.: Spatial C^2 PH quintic splines. In: Lyche, T., Mazure, M.–L., Schumaker, L.L. (eds.) Curve and Surface Design: Saint Malo 2002, pp. 147–156. Nashboro Press (2003)
9. Farouki, R.T., Neff, C.A.: Hermite interpolation by Pythagorean hodograph quintics. Math. Comp. 64, 1589–1609 (1995)
10. Fletcher, R.: Practical Methods of Optimization. Wiley, New York (1987)
11. Jüttler, B.: Hermite interpolation by Pythagorean hodograph curves of degree seven. Math. Comp. 70, 1089–1111 (2001)
12. Moon, H.P., Farouki, R.T., Choi, H.I.: Construction and shape analysis of PH quintic Hermite interpolants. Comput. Aided Geom. Design 18, 93–115 (2001)
13. Pelosi, F., Sampoli, M.L., Farouki, R.T., Manni, C.: A control polygon scheme for design of planar C^2 PH quintic spline curves. Comput. Aided Geom. Design 24, 28–52 (2007)
14. Sir, Z., Jüttler, B.: C^2 Hermite interpolation by Pythagorean hodograph space curves. Math. Comp. 76, 1373–1391 (2007)

Shape Curvatures of Planar Rational Spirals

Georgi H. Georgiev*

Faculty of Mathematics and Informatics, Konstantin Preslavsky University,
115 Universitetska Str. 9712 Shumen, Bulgaria

Abstract. We discuss geometric conditions for a planar rational curve to be a spiral. The main used tool for characterizing a spiral is the so-called shape curvature which is a differential-geometric invariant of planar curves with respect to the group of orientation-preserving similarities. Different formulas for computation of shape curvature are given. Rational curves representing spirals and circular arcs possess a natural description in terms of the shape curvature. This description is applied to a complete classification of quadratic Bézier spirals.

Keywords: shape curvature, affine transformations, direct similarities, planar rational spirals.

1 Introduction

Polynomial and rational planar spirals have many modeling applications. Various investigations of such spirals are connected with two points interpolations. The characterizing property of planar spirals, the monotony of the signed curvature, is always expressed in terms of the first derivative of the signed curvature. Both the signed curvature and its first derivative are invariant under Euclidean motions of the plane. But there is a larger group of transformations preserving the class of spirals. These transformations are direct similarities, which are affine transformations leaving invariant the orientation and the angles. The shape curvature is a differential-geometric invariant of planar curves with respect to the group of direct similarities. The purpose of this paper is to give an alternative description of planar spirals in terms of their shape curvatures. As an application, a complete classification of quadratic Bézier spirals is presented.

The paper is organized as follows. In the next section some preliminaries about affine transformations and signed curvature are given. After that the definition of the shape curvature and different formulas for its computation are stated. Section 4 is devoted to the characterization of spiral and circular arcs in terms of the shape curvature. In Section 5, the classification of quadratic Bézier spirals is examined by the use of the shape curvature. The paper concludes with some final remarks.

* The author is partially supported by Konstantin Preslavsky University of Shumen under grant No.RD-05-338/12.03.2010.

2 Affine Transformations and Signed Curvature

We start with basic facts concerning affine transformations of the plane and planar smooth curves. Matrix notations are a useful tool for investigations of affine images of planar curves.

2.1 Affine Transformations of the Plane

We assume that any point p in the Euclidean plane $\mathbb{R}^2$ is presented by its Cartesian coordinates x and y written as a column vector $\mathbf{p} = \begin{pmatrix} x \\ y \end{pmatrix}$. In other words we identify the point $p \in \mathbb{R}^2$ with its position vector $\mathbf{p}$. Every orientation-preserving affine transformation $\mathcal{A} : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ is given by

$$\bar{\mathbf{p}} = \mathbf{A}\mathbf{p} + \mathbf{b}\,, \tag{1}$$

where $\bar{\mathbf{p}} = \begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix}$ is the position vector of the point $\bar{p} = \mathcal{A}(p)$, $\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ is a real square matrix with $\det(\mathbf{A}) > 0$, and $\mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$ is a translation vector. These transformations form a group which is denoted by $Aff^{+}(\mathbb{R}^2)$

If $\mathbf{A}$ is an orthogonal matrix with $\det(\mathbf{A}) = 1$, then the affine transformation $\mathcal{A}$ is either a rotation or a translation. The rotations and translations of $\mathbb{R}^2$ form the well-known Euclidean motions group $E^{+}(\mathbb{R}^2)$ which is a subgroup of $Aff^{+}(\mathbb{R}^2)$. Consider the counterclockwise rotation $\mathcal{J} : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ given by

$$\bar{\mathbf{p}} = \mathbf{J}\mathbf{p}\,, \qquad \mathbf{J} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.$$

This rotation is called a complex structure of $\mathbb{R}^2$ because $\mathbf{J}^2\mathbf{p} = \mathbf{J}\mathbf{J}\mathbf{p} = -\mathbf{p}$ for any $\mathbf{p}$. Moreover, the equality

$$\mathbf{A}^T\mathbf{J}\mathbf{A} = \det(\mathbf{A})\mathbf{J}$$

holds for any matrix $\mathbf{A} \in \mathbb{R}^{2\times 2}$ with $\det(\mathbf{A}) \neq 0$.

If $\mathbf{A} = \lambda\mathbf{A}_o$, where λ is a positive real number and $\mathbf{A}_o$ is an orthogonal matrix with $\det(\mathbf{A}_o) = 1$, then $\|\mathbf{A}\mathbf{v}\| = \lambda\|\mathbf{v}\|$ for any vector $\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$. In this case the affine transformation $\mathcal{A}$ given by (1) is an orientation-preserving similarity transformation. Any such transformation is called a direct similarity with a similarity ratio λ. All direct similarities of the plane form a subgroup $Sim^{+}(\mathbb{R}^2)$ of the group $Aff^{+}(\mathbb{R}^2)$ and this subgroup is the smallest extension of the Euclidean motions group $E^{+}(\mathbb{R}^2)$.

2.2 Signed Curvature

Let $\mathcal{C} : I \longrightarrow \mathbb{R}^2$ be a three times differentiable regular curve defined on certain interval $I \subseteq \mathbb{R}$, and let

$$\mathbf{c}(t) = \big(x(t), y(t)\big)^T, \qquad t \in I \tag{2}$$

be a parametric representation of $\mathcal{C}$. Such a curve also is called a smooth planar curve of class C^3. This means that the three derivatives $\dot{\mathbf{c}}\ (t) = \frac{d}{dt}\big(x(t),\, y(t)\big)^T = \big(\ \dot{x}\ \ (t), \dot{y}\ \ (t)\big)^T$, $\ddot{\mathbf{c}}\ \ (t) = \frac{d^2}{dt^2}\big(x(t),\, y(t)\big)^T = \big(\ \ddot{x}\ \ (t), \ddot{y}\ \ (t)\big)^T$ and $\dddot{\mathbf{c}}\ \ (t) = \frac{d^3}{dt^3}\big(x(t),\, y(t)\big)^T = \big(\ \dddot{x}\ \ (t), \dddot{y}\ \ (t)\big)^T$ exist and $\dot{\mathbf{c}}\ (t) \neq= (0,0)^T$ for any $t \in I$. The signed curvature of the curve c is the real-valued function given by

$$K_c(t) = \frac{\langle \ddot{\mathbf{c}}\ (t)\,,\, \mathbf{J}\ \dot{\mathbf{c}}\ (t)\rangle}{\langle \dot{\mathbf{c}}\ (t)\,,\, \dot{\mathbf{c}}\ (t)\rangle^{3/2}} = \frac{\dot{x}\ (t)\ \ddot{y}\ (t) -\ \dot{y}\ (t)\ \ddot{x}\ (t)}{\Big(\big(\ \dot{x}\ (t)\big)^2 + \big(\ \dot{y}\ (t)\big)^2\Big)^{3/2}}\,, \quad t \in I \tag{3}$$

where $\langle .\,,\, . \rangle$ denotes the scalar product of two vectors. On the other hand, the scalar product of the vectors $\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$ and $\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$ can be written in the form $\langle \mathbf{v}\,,\, \mathbf{w}\rangle = \mathbf{v}^T\ \cdot\ \mathbf{w} = (v_1, v_2)\ \cdot\ \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$, where the symbol " $\cdot$ " denotes the usual matrix multiplication of a vector-row and a vector-column. Then, the signed curvature can be evaluated by the alternative formula

$$K_{\mathcal{C}} = \frac{\ddot{\mathbf{c}}^{\,T}\ \cdot \mathbf{J}\ \dot{\mathbf{c}}}{\big(\dot{\mathbf{c}}^{\,T} \cdot\ \dot{\mathbf{c}}\ \big)^{3/2}}\,, \tag{4}$$

in which the parameter t is omitted for brevity. The first derivative of $K(t)$ is used in many investigations connected to spirals. From formulas (3) and (4) it follows that

$$\begin{aligned} \frac{d}{dt}K_c &= \frac{\big\langle\ \dddot{\mathbf{c}}\,,\, \mathbf{J}\ \dot{\mathbf{c}}\ \big\rangle \big\langle\ \dot{\mathbf{c}}\,,\, \dot{\mathbf{c}}\ \big\rangle - 3\big\langle\ \ddot{\mathbf{c}}\,,\, \mathbf{J}\ \dot{\mathbf{c}}\ \big\rangle \big\langle\ \ddot{\mathbf{c}}\,,\, \dot{\mathbf{c}}\ \big\rangle}{\big\langle\ \dot{\mathbf{c}}\,,\, \dot{\mathbf{c}}\ \big\rangle^{5/2}} \\ &= \frac{\big(\ \dddot{\mathbf{c}}^{\,T}\ \cdot \mathbf{J}\ \dot{\mathbf{c}}\ \big)\big(\dot{\mathbf{c}}^{\,T} \cdot\ \dot{\mathbf{c}}\ \big) - 3\big(\ \ddot{\mathbf{c}}^{\ T}\ \cdot \mathbf{J}\ \dot{\mathbf{c}}\ \big)\big(\ddot{\mathbf{c}}^{\,T} \cdot\ \dot{\mathbf{c}}\ \big)}{\big(\dot{\mathbf{c}}^{\,T} \cdot\ \dot{\mathbf{c}}\ \big)^{5/2}}\,. \end{aligned} \tag{5}$$

Let $\mathcal{A}\ :\ \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ be an affine transformation given by (1). Then the image $\bar{\mathcal{C}} = \mathcal{A}(\mathcal{C})$ is a regular curve with a parametrization

$$\bar{\mathbf{c}}\ (t) = \mathbf{A}\mathbf{c}(t) + \mathbf{b}\,, \qquad t \in I\,. \tag{6}$$

Consequently, the derivatives of $\bar{\mathbf{c}}\ (t)$ are $\dot{\bar{\mathbf{c}}}\ (t) = \mathbf{A}\ \dot{\mathbf{c}}\ (t)$, $\ddot{\bar{\mathbf{c}}}\ (t) = \mathbf{A}\ \ddot{\mathbf{c}}\ (t)$ and $\dddot{\bar{\mathbf{c}}}\ (t) = \mathbf{A}\ \dddot{\mathbf{c}}\ (t)$. From here, it follows that the signed curvature of $\bar{\mathcal{C}}$ is

$$K_{\bar{\mathcal{C}}} = \frac{\ddot{\mathbf{c}}^{\,T}\ \cdot \det(\mathbf{A})\mathbf{J}\ \dot{\mathbf{c}}}{\big(\ \dot{\mathbf{c}}^{\ T}\ \cdot\ \mathbf{A}^T\mathbf{A}\ \dot{\mathbf{c}}\ \big)^{3/2}}\,. \tag{7}$$

If $\mathcal{A} : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ is a Euclidean motion then the equalities $K_{\bar{\mathcal{C}}}(t) = K_{\mathcal{C}}(t)$ and $\frac{d}{dt}K_{\bar{\mathcal{C}}}(t) = \frac{d}{dt}K_{\mathcal{C}}(t)$ hold for any $t \in I$. If $\mathcal{A} : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ is a a direct similarity with a similarity ratio $\lambda > 0$, then by (7) the equalities $K_{\bar{\mathcal{C}}}(t) = \frac{1}{\lambda}K_{\mathcal{C}}(t)$ and $\frac{d}{dt}K_{\bar{\mathcal{C}}}(t) = \frac{1}{\lambda}\frac{d}{dt}K_{\mathcal{C}}(t)$ are fulfilled for any $t \in I$.

3 Shape Curvature and Its Computation

As in the previous section we consider a three times differentiable curve with a parametrization (2).

Definition 1. *Let $\mathcal{C} : I \longrightarrow \mathbb{R}^2$ be a curve of class C^3 with nonzero signed curvature $K_{\mathcal{C}}(t)$. A shape curvature of $\mathcal{C}$ is a real-valued function $\widetilde{K}_{\mathcal{C}}(t)$ defined by*

$$\widetilde{K}_{\mathcal{C}}(t) = \frac{1}{\| \dot{\mathbf{c}}\ (t)\|}\frac{d}{dt}\Big(\frac{1}{K_{\mathcal{C}}(t)}\Big) = -\frac{1}{\sqrt{\langle \dot{\mathbf{c}}\ (t)\, ,\, \dot{\mathbf{c}}\ (t)\rangle}\,(K_{\mathcal{C}}(t))^2}\frac{d}{dt}K_{\mathcal{C}}(t). \quad (8)$$

Using (3), (4) and (5) we obtain

$$\widetilde{K}_{\mathcal{C}}(t) = \frac{3\big\langle\, \ddot{\mathbf{c}}\, , \mathbf{J}\,\dot{\mathbf{c}}\,\big\rangle\big\langle\, \ddot{\mathbf{c}}\, , \dot{\mathbf{c}}\,\big\rangle - \big\langle\, \dddot{\mathbf{c}}\, , \mathbf{J}\,\dot{\mathbf{c}}\,\big\rangle\big\langle\, \dot{\mathbf{c}}\, , \dot{\mathbf{c}}\,\big\rangle}{\big\langle\, \ddot{\mathbf{c}}\, , \mathbf{J}\,\dot{\mathbf{c}}\,\big\rangle^2}\,, \quad (9)$$

or equivalently in matrix notations,

$$\widetilde{K}_{\mathcal{C}}(t) = \frac{3(\,\ddot{\mathbf{c}}^{\,T}\cdot\mathbf{J}\,\dot{\mathbf{c}}\,)(\ddot{\mathbf{c}}^{T}\cdot\,\dot{\mathbf{c}}\,) - (\,\dddot{\mathbf{c}}^{\,T}\cdot\mathbf{J}\,\dot{\mathbf{c}}\,)(\dot{\mathbf{c}}^{T}\cdot\,\dot{\mathbf{c}}\,)}{\big(\ddot{\mathbf{c}}^{T}\cdot\mathbf{J}\,\dot{\mathbf{c}}\,\big)^2}\,. \quad (10)$$

Example 1. If the curve is a circle, then by (8) its shape curvature is identically zero.

Example 2. The logarithmic spiral given by $\mathbf{c}(t) = \big(e^t\cos(t)\, ,\, e^t\sin(t)\big)^T$ has a shape curvature which is equal to 1 for any t.

Now, we will study the relations between the affine transformations and the shape curvature.

Proposition 1. *Any direct similarity leaves invariant the shape curvatures of the planar curves.*

Proof. Let $\mathcal{A} : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ be a direct similarity with a similarity ratio $\lambda > 0$. Suppose that $\mathcal{C} : I \longrightarrow \mathbb{R}^2$ is a curve of class C^3 with nonzero curvature $K_{\mathcal{C}}(t)$ for any $t \in I$. Then, the affine image $\bar{\mathcal{C}} = \mathcal{A}(\mathcal{C})$ is a three times differentiable curve with the shape curvature

$$\widetilde{K}_{\bar{\mathcal{C}}}(t) = \frac{1}{\| \dot{\bar{\mathbf{c}}}\ (t)\|}\frac{d}{dt}\Big(\frac{1}{K_{\bar{\mathcal{C}}}(t)}\Big) = \frac{1}{\lambda\| \dot{\mathbf{c}}\ (t)\|}\frac{d}{dt}\Big(\frac{\lambda}{K_{\mathcal{C}}(t)}\Big) = \widetilde{K}_{\mathcal{C}}(t)\,. \qquad \square$$

Locally, a planar curve of class C^3 is determined uniquely by its shape curvature up to a direct similarity. The proof of this assertion can be found in [4] and [5]. For the unit-speed curve $\mathcal{C}$ with arc-length parameter s we have

$$\widetilde{K}_{\mathcal{C}}(s) = \frac{d}{ds}\Big(\frac{1}{K_{\mathcal{C}}(s)}\Big) = -\,\frac{1}{\big(K_{\mathcal{C}}(s)\big)^2}\,\frac{d}{ds}K_{\mathcal{C}}(s).$$

In other words, the ratio $\dfrac{\frac{d}{ds}K_{\mathcal{C}}(s)}{\big(K_{\mathcal{C}}(s)\big)^2}$ also is a differential geometric invariant with respect to the group $Sim^+(\mathbb{R}^2)$. This invariant is considered in [1].

Now we may obtain a common formula for the shape curvature of an affine image of a planar curve.

Theorem 1. *Let $\mathcal{A} : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ be an affine transformation given by (1). Consider a curve $\mathcal{C}$ with parametrization (2) and its affine image $\bar{\mathcal{C}} = \mathcal{A}(\mathcal{C})$ with a parametrization (6). Then, the shape curvature of $\bar{\mathcal{C}}$ is*

$$\widetilde{K}_{\bar{\mathcal{C}}}(t) = \frac{3\big(\,\ddot{\mathbf{c}}^{\,T}\cdot\mathbf{J}\,\dot{\mathbf{c}}\,\big)\big(\ddot{\mathbf{c}}^{\,T}\cdot\mathbf{A}^T\mathbf{A}\,\dot{\mathbf{c}}\,\big) - \big(\,\dddot{\mathbf{c}}^{\,T}\cdot\mathbf{J}\,\dot{\mathbf{c}}\,\big)\big(\dot{\mathbf{c}}^{\,T}\cdot\mathbf{A}^T\mathbf{A}\,\dot{\mathbf{c}}\,\big)}{\det(\mathbf{A})\,\big(\ddot{\mathbf{c}}^{\,T}\cdot\mathbf{J}\,\dot{\mathbf{c}}\,\big)^2}\,. \qquad (11)$$

Proof. As in previous section, the derivatives of $\bar{\mathbf{c}}\,(t)$ are

$$\frac{d^i}{dt^i}\,\bar{\mathbf{c}}\,(t) = \mathbf{A}\frac{d^i}{dt^i}\mathbf{c}(t), \qquad i = 1,2,3\,.$$

For any two vectors $\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$ and $\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$ and for any matrix $\mathbb{A} \in \mathbf{R}^{2\times 2}$ with $\det(\mathbb{A}) \neq 0$, the equalities

$$\big(\mathbb{A}\mathbf{v}\big)^T\cdot\mathbb{A}\mathbf{w} = \mathbf{v}^T\cdot\mathbb{A}^T\mathbb{A}\mathbf{w} \quad \text{and} \quad \big(\mathbb{A}\mathbf{v}\big)^T\cdot\mathbf{J}\mathbb{A}\mathbf{w} = \mathbf{v}^T\cdot\det(\mathbb{A})\mathbf{J}\mathbf{w}$$

hold. Applying these equations and the matrix formula (10) we can express the required shape curvature as

$$\widetilde{K}_{\bar{\mathcal{C}}}(t) = \frac{3\big(\,\ddot{\mathbf{c}}^{\,T}\cdot\det(\mathbf{A})\mathbf{J}\,\dot{\mathbf{c}}\,\big)\big(\ddot{\mathbf{c}}^{\,T}\cdot\mathbf{A}^T\mathbf{A}\,\dot{\mathbf{c}}\,\big)}{\big(\,\ddot{\mathbf{c}}^{\,T}\cdot\det(\mathbf{A})\,\mathbf{J}\,\dot{\mathbf{c}}\,\big)^2} - \frac{\big(\,\dddot{\mathbf{c}}^{\,T}\cdot\det(\mathbf{A})\mathbf{J}\,\dot{\mathbf{c}}\,\big)\big(\dot{\mathbf{c}}^{\,T}\cdot\mathbf{A}^T\mathbf{A}\,\dot{\mathbf{c}}\,\big)}{\big(\,\ddot{\mathbf{c}}^{\,T}\cdot\det(\mathbf{A})\,\mathbf{J}\,\dot{\mathbf{c}}\,\big)^2}\,.$$

Finally, eliminating the common factor $\det(\mathbf{A})$ in the numerator and the denominator we get the formula (11). $\square$

4 Characterizing the Rational Spirals by Their Shape Curvatures

Planar rational curves are widely used in Computer Aided Geometric Design. One topic in this area is the investigation and applications of rational spirals (see e.g. [2], [3], [7], [9], [10] and [13]). Recently, planar rational cubics representing circular arcs have been introduced for geometric Hermite interpolation (see [6] and [14]). Both rational spirals and rational curves representing circular arcs have a natural description in terms of their shape curvatures.

In this section we consider a rational curve $\mathcal{C} : [0,1] \longrightarrow \mathbb{R}^2$ which is given by a parametrization (2), i.e. we assume that the coordinate functions $x(t)$ and $y(t)$ are rational. Such a curve with monotone curvature is called a planar rational spiral. Moreover, for planar rational curves we introduce the following functions

$$\begin{aligned} Q_1(t) &= \langle \dot{\mathbf{c}}\,(t)\,, \dot{\mathbf{c}}\,(t)\rangle &&= \left(\dot{\mathbf{c}}^{\,T} \cdot \dot{\mathbf{c}}\,\right), \\ Q_2(t) &= \langle \ddot{\mathbf{c}}\,(t)\,, \dot{\mathbf{c}}\,(t)\rangle &&= \left(\ddot{\mathbf{c}}^{\,T} \cdot \dot{\mathbf{c}}\,\right), \\ Q_3(t) &= \langle \ddot{\mathbf{c}}\,(t)\,, \mathbf{J}\,\dot{\mathbf{c}}\,(t)\rangle &&= \left(\,\ddot{\mathbf{c}}^{\;T} \cdot \mathbf{J}\,\dot{\mathbf{c}}\,\right), \\ Q_4(t) &= \langle \dddot{\mathbf{c}}\,(t)\,, \mathbf{J}\,\dot{\mathbf{c}}\,(t)\rangle &&= \left(\,\dddot{\mathbf{c}}^{\;T} \cdot \mathbf{J}\,\dot{\mathbf{c}}\,\right). \end{aligned}$$

Theorem 2. *Let $\mathcal{C} : [0,1] \longrightarrow \mathbb{R}^2$ be a three times differentiable curve with nonzero curvature. Then the following four statements are equivalent:*

(a) *The rational function $f(t) = 3Q_2(t)Q_3(t) - Q_1(t)Q_4(t)$ does not change its sign in the interval* [0,1].
(b) *The shape curvature of $\mathcal{C}$ does not change its sign in the interval* [0,1].
(c) *The first derivative of the curvature of $\mathcal{C}$ does not change its sign in the interval* [0,1].
(d) *The curve $\mathcal{C}$ is a rational spiral.*

Proof. The equivalence (a)$\Leftrightarrow$(b) follows from the formula (9). According to Definition 1., the shape curvature and the derivative of the signed curvature have opposite signs. This implies immediately (b)$\Leftrightarrow$(c). The equivalence (c)$\Leftrightarrow$(d) is obvious. □

In the same way we can describe rational curves representing circular arcs.

Corollary 1. *For a three times differentiable rational curve $\mathcal{C} : [0,1] \longrightarrow \mathbb{R}^2$ with nonzero curvature, the following four statements are equivalent:*

(a) *The rational function $f(t) = 3Q_2(t)Q_3(t) - Q_1(t)Q_4(t)$ is identically zero in the interval* [0,1].
(b) *The shape curvature of $\mathcal{C}$ vanishes everywhere.*
(c) *The curvature function of $\mathcal{C}$ is constant .*
(d) *The curve $\mathcal{C}$ is a circular arc.*

Now, combining Theorem 1 and Theorem 2 we can formulate the main result of the paper.

Corollary 2. *The largest subgroup of the affine group $Aff^+(\mathbb{R}^2)$ whose transformations preserves the class of rational spirals is the group of direct similarities $Sim^+(\mathbb{R}^2)$.*

We observe that the functions $f(t)$ and $\frac{d}{dt}K_{\mathcal{C}}(t)$ are not invariant under direct similarities different from Euclidean motions. Hence, the statement (b) in Theorem 2 is the determining property for rational spirals. Note that the rational function $f(t)$ can be used for a fast detection of rational spirals and circular arcs without computing the shape curvature or the signed curvature. Other properties of spirals are considered in [11].

5 A Classification of Quadratic Bézier Spirals

Quadratic Bézier spirals are studied in [7] and [12] by the use of the first derivative of the signed curvature. Our approach based on the shape curvature gives more clear and accurate description of these spirals.

Let $\mathcal{B} : [0,1] \longrightarrow \mathbb{R}^2$ be a quadratic Bézier curve defined by

$$\mathbf{b}(t) = (1-t)^2\mathbf{p}_0 + 2(1-t)t\,\mathbf{p}_1 + t^2\,\mathbf{p}_2\,, \qquad t \in [0,1], \tag{12}$$

where the control points $\mathbf{p}_0$, $\mathbf{p}_1$ and $\mathbf{p}_2$ are non-collinear. In a previous author's paper [8] it is shown that the shape curvature function of the curve $\mathcal{B}$ given by (12) is the following non-constant linear function

$$\widetilde{K}_{\mathcal{B}}(t) = 3\,\frac{\langle \Delta^2\mathbf{p}_0\,,\,\Delta^1\mathbf{p}_0\rangle}{\langle \Delta^2\mathbf{p}_0\,,\,\mathbf{J}\Delta^1\mathbf{p}_0\rangle} + 3\,\frac{\langle \Delta^2\mathbf{p}_0\,,\,\Delta^2\mathbf{p}_0\rangle}{\langle \Delta^2\mathbf{p}_0\,,\,\mathbf{J}\Delta^1\mathbf{p}_0\rangle}\,t\,, \quad t \in [0,1], \tag{13}$$

where $\Delta^1\mathbf{p}_0 = \mathbf{p}_1 - \mathbf{p}_0$ and $\Delta^2\mathbf{p}_0 = \mathbf{p}_0 - 2\mathbf{p}_1 + \mathbf{p}_2$. This implies that all quadratic Bézier spirals can be classified according to the values of the shape curvature at the end points.

Theorem 3. *Let a and $b \neq 0$ be real numbers and let $\mathcal{B} : [0,1] \longrightarrow \mathbb{R}^2$ be a quadratic Bézier curve with control points $\mathbf{p}_0 = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$, $\mathbf{p}_1 = \begin{pmatrix} a \\ b \end{pmatrix}$ and $\mathbf{p}_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. Assume that the shape curvature function of $\mathcal{B}$ is denoted by $\widetilde{K}_{\mathcal{B}}(t)$ and $\sigma_0 = \widetilde{K}_{\mathcal{B}}(0)$, $\sigma_2 = \widetilde{K}_{\mathcal{B}}(1)$. Then the considered curve $\mathcal{B}$ is a spiral if and only if just one of the following four conditions is fulfilled:*

(i) $0 \leq \sigma_0 < \sigma_2$, *or equivalently,* $a < 0$, $b > 0$ *and* $(a + \frac{1}{2})^2 + b^2 < \frac{1}{4}$.
(ii) $0 \leq \sigma_2 < \sigma_0$, *or equivalently,* $a > 0$, $b < 0$ *and* $(a - \frac{1}{2})^2 + b^2 < \frac{1}{4}$.
(iii) $\sigma_0 < \sigma_2 \leq 0$, *or equivalently,* $a > 0$, $b > 0$ *and* $(a - \frac{1}{2})^2 + b^2 < \frac{1}{4}$.
(iv) $\sigma_2 < \sigma_0 \leq 0$, *or equivalently,* $a < 0$, $b < 0$ *and* $(a + \frac{1}{2})^2 + b^2 < \frac{1}{4}$.

Proof. We can express the relations between the pairs $a\,,\,b$ and $\sigma_0\,,\,\sigma_2$ in an explicit form. First, suppose that the numbers a and $b \neq 0$ are given. Then, $\Delta^2\mathbf{p}_0 = \begin{pmatrix} -2a \\ -2b \end{pmatrix}$, $\Delta^1\mathbf{p}_0 = \begin{pmatrix} a+1 \\ b \end{pmatrix}$, and by (13) the shape curvature is

$$\widetilde{K}_{\mathcal{B}}(t) = 3\,\frac{a^2 + b^2 - a}{b} - 3\,\frac{2a^2 + 2b^2}{b}\,t\,, \quad t \in [0,1]. \tag{14}$$

From here it follows that

$$\widetilde{K}_{\mathcal{B}}(0) = \sigma_0 = \frac{3a^2 + 3b^2 - 3a}{b}, \qquad \widetilde{K}_{\mathcal{B}}(1) = \sigma_2 = \frac{-3a^2 - 3b^2 - 3a}{b}. \tag{15}$$

Secondly, suppose that the shape curvatures σ_0 and σ_2 at the endpoints are given. Then the equalities (15) may be written as

$$\left| \begin{array}{l} 3a^2 + 3b^2 - 3a = b\sigma_0 \\ -3a^2 - 3b^2 - 3a = b\sigma_2\,. \end{array} \right.$$

Solving the last system with respect to a and b provide the coordinates of the control point $\mathbf{p}_1$

$$a = \frac{\sigma_0^2 - \sigma_2^2}{(\sigma_0 + \sigma_2)^2 + 36}\,, \qquad b = \frac{6(\sigma_2 - \sigma_0)}{(\sigma_0 + \sigma_2)^2 + 36}\,. \tag{16}$$

Now, using Theorem 2 we can describe the four cases in which the quadratic Bézier curve will be a spiral.

Case (i). The shape curvature function (14) is positive and increasing, i.e. $0 \leq \sigma_0 < \sigma_2$. By (16) we have $a < 0$ and $b > 0$. Applying (15) we obtain

$$a^2 + b^2 - a \geq 0 \qquad \text{and} \qquad -a^2 - b^2 - a > 0,$$

or

$$\left(a - \frac{1}{2}\right)^2 + b^2 \geq \frac{1}{4} \qquad \text{and} \qquad \left(a + \frac{1}{2}\right)^2 + b^2 < \frac{1}{4}.$$

This means that the point $\mathbf{p}_1 = \begin{pmatrix} a \\ b \end{pmatrix}$ belongs to the open semi-disk

$$D_1^+ = \left\{ a < 0,\ b > 0,\ \left(a + \frac{1}{2}\right)^2 + b^2 < \frac{1}{4} \right\}.$$

Conversely, if $\mathbf{p}_1 \in D_1^+$, then from (15) it follows that $0 \leq \sigma_0$ and $0 < \sigma_2$. By (16) and $b > 0$ we have $\sigma_0 < \sigma_2$. Hence, $0 \leq \sigma_0 < \sigma_2 \Leftrightarrow \mathbf{p}_1 \in D_1^+$.
In a similar way we obtain the remaining three cases:

Case (ii). The shape curvature function (14) is non-negative and decreasing $\Leftrightarrow$ $0 \leq \sigma_2 < \sigma_0$ $\Leftrightarrow$ the point $\mathbf{p}_1$ belongs to the open semi-disk

$$D_2^- = \left\{ a > 0,\ b < 0,\ \left(a - \frac{1}{2}\right)^2 + b^2 < \frac{1}{4} \right\}.$$

Case (iii). The shape curvature function (14) is non-positive and increasing $\Leftrightarrow$ $\sigma_0 < \sigma_2 \leq 0$ $\Leftrightarrow$ the point $\mathbf{p}_1$ belongs to the open semi-disk

$$D_2^+ = \left\{ a > 0,\ b > 0,\ \left(a - \frac{1}{2}\right)^2 + b^2 < \frac{1}{4} \right\}.$$

Case (iv). The shape curvature function (14) is non-positive and decreasing $\Leftrightarrow \quad \sigma_2 < \sigma_0 \ \leq 0 \Leftrightarrow$ the point $\mathbf{p}_1$ belongs to the open semi-disk

$$D_1^- = \left\{ a < 0,\ b < 0,\ \left(a + \frac{1}{2}\right)^2 + b^2 < \frac{1}{4} \right\}. \ \square$$

Corollary 3. *Let* $\mathbf{q}_0 = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$ *and* $\mathbf{q}_2 = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix}$ *be two different points in the Euclidean plane. Suppose that two different real numbers* σ_0, σ_2 *are given and they are with the same sign or one of them is zero. Then, there exists a unique quadratic Bézier spiral with end points* $\mathbf{q}_0$ *and* $\mathbf{q}_2$ *such that the values of its shape curvature function at* $\mathbf{q}_0$ *and* $\mathbf{q}_2$ *are* σ_0 *and* σ_2*, respectively.*

Proof. First, we assume that $\mathbf{q}_0 = \mathbf{p}_0 = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$ and $\mathbf{q}_2 = \mathbf{p}_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. According to the proof of Theorem 3 there exists a unique point

$$\mathbf{p}_1 = \begin{pmatrix} \dfrac{\sigma_0^2 - \sigma_2^2}{(\sigma_0 + \sigma_2)^2 + 36} \\ \dfrac{6(\sigma_2 - \sigma_0)}{(\sigma_0 + \sigma_2)^2 + 36} \end{pmatrix}$$

such that the quadratic Bézier curve $\mathcal{B}$ with control points $\mathbf{p}_0$, $\mathbf{p}_1$ and $\mathbf{p}_2$ have the required properties.

Second, we consider the general case. Then, there is a unique direct similarity $\mathcal{A} : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ which transforms the ordered pair $\mathbf{p}_0$, $\mathbf{p}_2$ into the ordered pair $\mathbf{q}_0$, $\mathbf{q}_2$. This similarity can be written in matrix form as

$$\begin{pmatrix} \overline{x} \\ \overline{y} \end{pmatrix} = \begin{pmatrix} \frac{1}{2}(x_2 - x_0) & -\frac{1}{2}(y_2 - y_0) \\ \frac{1}{2}(y_2 - y_0) & \frac{1}{2}(x_2 - x_0) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \frac{1}{2}(x_0 + x_2) \\ \frac{1}{2}(y_0 + y_2) \end{pmatrix}.$$

If the same transformation $\mathcal{A}$ maps the point $\mathbf{p}_1$ to a point $\mathbf{q}_1 \in \mathbb{R}^2$, then the image $\overline{\mathcal{B}} = \mathcal{A}(\mathcal{B})$ is a quadratic Bézier curve with control points $\mathbf{q}_0$, $\mathbf{q}_1$ and $\mathbf{q}_2$. Since both curves $\overline{\mathcal{B}}$ and $\mathcal{B}$ have the same shape curvature function, the curve $\overline{\mathcal{B}}$ is a spiral with shape curvatures σ_0 and σ_2 at the end points. $\square$

6 Conclusion

In this paper we show that the planar rational spirals are closely related to direct similarities of the plane. First, curvature monotony conditions is expressed in terms of shape curvature. Then, any direct similarity preserves the class of the rational spirals and its subclass of rational curves representing circular arcs. Finally, all quadratic Bézier spirals are completely described by the use of the shape curvature.

Acknowledgments. Part of this paper has been written during an author's research visit at Institut für Mathematik, Technische Universität Berlin in September 2010. The author is grateful to Prof. Udo Simon and TU-Berlin for their hospitality and support.

References

1. Chou, K.-S., Qu, C.: Motions of Curves in Similarity Geometries and Burgers-mKdV Hierarchies. Chaos, Solitons and Fractals 19, 47–53 (2004)
2. Dietz, D., Piper, B.: Interpolation with Cubic Spirals. Comp. Aided Geom. Design 21, 165–180 (2004)
3. Dietz, D., Piper, B., Sebe, E.: Rational Cubic Spirals. Computer-Aided Design 40, 3–12 (2008)
4. Encheva, R., Georgiev, G.: Curves on the Shape Sphere. Results in Mathematics 44, 279–288 (2003)
5. Encheva, R.P., Georgiev, G.H.: Similar Frenet Curves. Results in Mathematics 55, 359–372 (2009)
6. Farin, G.: Geometric Hermite Interpolation with Circular Precision. Computer-Aided Design 40, 476–479 (2008)
7. Frey, W.H., Field, D.A.: Designing Bézier Conic Segments with Monotone Curvature. Comp. Aided Geom. Design 17, 457–483 (2000)
8. Georgiev, G.H.: Shapes of Blane Bézier Curves. In: Chenin, P., Lyche, T., Schumaker, L.L. (eds.) Curve and Surface Design: Avignon 2006, pp. 143–152. Nashboro Press, Brentwood TN (2007)
9. Goodman, T.N.T., Meek, D.S.: Planar Interpolation with a Pair of Rational Spirals. Journal of Computational and Applied Mathematics 201, 112–125 (2007)
10. Goodman, T.N.T., Meek, D.S., Walton, D.J.: An Involute Spiral that Matches G2 Hermite Data in the Plane. Comp. Aided Geom. Design 26, 733–756 (2009)
11. Kurnosenko, A.I.: General Properties of Spiral Plane Curves. Journal of Mathematical Sciences 161, 405–418 (2009)
12. Kurnosenko, A.I.: Applying Inversion to Construct Planar, Rational, Spiral Curves that Satisfy Two-point G2 Hermite Data. Comp. Aided Geom. Design 27, 262–280 (2010)
13. Li, Z., Ma, L., Meek, D., Tan, W., Mao, Z., Zhao, M.: Curvature Monotony Condition for Rational Quadratic B-spline Curves. In: Gavrilova, M.L., Gervasi, O., Kumar, V., Tan, C.J.K., Taniar, D., Laganá, A., Mun, Y., Choo, H. (eds.) ICCSA 2006. LNCS, vol. 3980, pp. 1118–1126. Springer, Heidelberg (2006)
14. Walton, D.J., Meek, D.S.: G2 Hermite Interpolation with Circular Precision. Computer-Aided Design 42, 749–758 (2010)

Volumetric Geometry Reconstruction of Turbine Blades for Aircraft Engines

David Großmann[1] and Bert Jüttler[2]

[1] MTU Aero Engines GmbH, Munich, Germany
[2] Johannes Kepler University, Institute of Applied Geometry, Linz, Austria

Abstract. We present a framework for generating a trivariate B-spline parametrization of turbine blades from measurement data generated by optical scanners. This new representation replaces the standard patch-based representation of industrial blade designs. In a first step, the blade surface is represented by a smoothly varying family of B-spline curves. In a second step, the blade is parametrized by a trivariate B-spline volume. The resulting model is suitable for numerical simulation via isogeometric analysis, as well as for a fully automatic structured mesh generation with standard finite elements. We focus on the industrial applicability of the framework, by using standard turbine blade features throughout the process.

Keywords: volumetric geometry reconstruction, turbine blades, trivariate B-Splines, numerical simulation, isogeometric analysis.

1 Introduction

The commercial activities of MTU Aero Engines focus on developing, manufacturing and repairing turbine engines for aircrafts. The volumetric B-spline parametrization – which is discussed in the present paper – enables us to explore new approaches to the challenging tasks of high-quality and efficient numerical simulations of turbine blades.

First, the new numerical simulation approach of isogeometric analysis (IGA), introduced by Hughes et al. [10], uses geometry mappings represented by volumetric NURBS parametrizations, and the finite-dimensional spaces needed for the Galerkin projection are defined with the help of these parametrizations. As a major advantage, only one representation of the blade geometry is required, which is used throughout the entire process of design, simulation, and manufacturing. Consequently, the geometrical errors introduced by approximating the blade geometry by finite element meshes are eliminated and the number of unknowns at the coarsest discretization level is significantly decreased. This new approach seems to be particularly well suited for blade geometries, since they are less complex than general CAD models.

Second, the volumetric description can be used for automatically partitioning the blade into several volumetric patches. Consequently, a fully automatic structured mesh generation for standard finite elements becomes possible.

B-spline volumes have been discussed in the classical literature in Computer Aided Design, e.g see [9]. The existing literature on volume parametrizations by B-splines concentrates mostly on applications to object modeling via free-from deformations. More recently, the construction of B-spline volume parametrizations for isogeometric analysis was discussed. The paper [2] designs B-spline volumes by sweeping and uses them for isogeometric analysis. In [14], the authors use harmonic functions to generate a spline parametrization of general cylinder-type objects. The regularization of B-spline volumes is addressed in [18].

With the focus on the industrial applicability, we present a process that generates a volumetric B-spline parametrization of a turbine blade. Our approach is fully compatible with the natural process flow in turbine blade engineering. Furthermore we can cover the industrial standards and requirements of a geometric blade model, and the intermediate results and methods are useful for performing standard CAD and CAE operations.

After presenting an outline of our approach, the paper describes the steps needed for the volumetric model generation. We conclude with some final remarks.

2 Problem Specification and Outline

We assume a triangular mesh of a turbine blade, generated by an optical measurement system or by sampling a standard (surface) CAD description of the blade. Therefore, our framework is usable for the integration of physical objects and geometric models.

Based on the mesh representation of the blade, we generate a volumetric tensor-product B-spline model of the blade by an almost automatic geometry reconstruction process, which requires only very little interaction with the user. The surface parametrization of the resulting model is suitable for a new parametrization of the blade surface which replaces the traditional blade parametrization with multiple trimmed surface patches. The volume parametrization of the model is suitable for high-quality numerical simulations using the isogeometric analysis or by a fully automatic structured mesh generation with standard finite elements.

Fig. 1 shows a disk with blades in a modern turbine, the standard position of a single blade and the typical shape of a blade. For the sake of brevity we shall denote both turbine blades and compressor blades as turbine blades or blades only.

The remainder of this section summarizes the proposed modeling framework. Starting with a triangular mesh $\mathcal{M}$ of a blade as input, we generate a volumetric tensor-product B-spline model in the following steps:

1. The user defines the number of slicing surfaces and the desired number of degrees of freedom for the tensor-product B-spline volume.
2. We generate two families of slicing surfaces $\mathcal{F}_\alpha$ and $\mathcal{F}_\beta$ which intersect the blade in well-behaved slices $\mathcal{C}_\omega$ between the blade boundary and the tip.

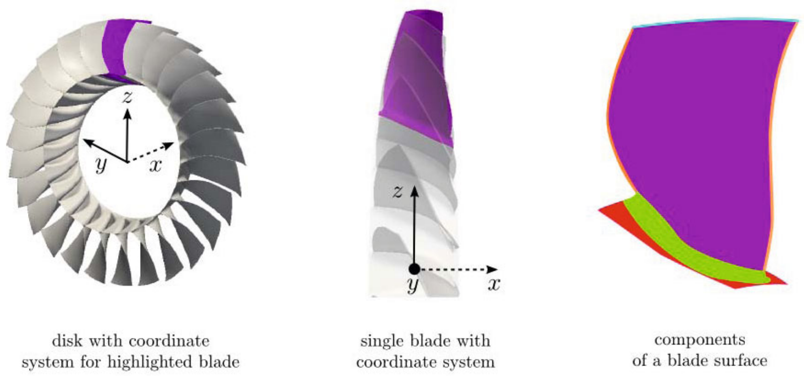

z
y
x
disk with coordinate system for highlighted blade
z
y
x
single blade with coordinate system
components of a blade surface

Fig. 1. From left to right: A disk with blades of a modern turbine engine. For our framework we assume the standard position of a disk, where the x-axis equals the turbine axis in the direction of the current. Considering a single blade, the z-axis equals the radial direction and the y-axis completes the right-handed orthonormal system. The shape of a blade consists of several parts, the endwall (red), the fillet (green), the airfoil with side parts (magenta) and edge parts (orange), and the tip (light blue).

3. We segment the slices $\mathcal{C}_\omega$ into edge and side parts, preparing them for the generation of a volumetric tensor-product B-spline model and considering the blade topology. For the airfoil part, the transitions between the four parts are called *wedge points.*
4. We fit the slices $\mathcal{C}_\omega$ by B-spline curves $\mathbf{c}_\omega(u)$ with identical degrees d and knot vectors $\mathcal{U}$. We use an iterative process to optimize the parametrization. The parameters associated with the wedge points (which we call *wedge knots*), however, are kept fixed.
5. We split the closed B-spline curves $\mathbf{c}_\omega(u)$ at the wedge knots into four boundary curves and generate surfaces $\mathbf{s}_\omega(u, v)$ by extending the boundary curves bilinearly to the inner part of the blade using Coons patches.
6. We interpolate the surfaces $\mathbf{s}_\omega(u, v)$ in sweeping direction w for a volumetric tensor-product B-spline model $\mathbf{v}(u, v, w)$ of the blade. An additional B-spline block at the bottom completes the model.

3 Slicing Surfaces

We represent the slicing surfaces $\mathcal{F}_\alpha$ and $\mathcal{F}_\beta$ as implicitly defined algebraic surfaces. This allows us to use their advantages compared to parametric surfaces, e.g. no data parametrization is needed for fitting processes and relatively simple algorithms for computing intersections with meshes and for blends are available.

An algebraic spline surface $\mathcal{F}$ is defined as the zero level set of a tensor-product spline function of (tri-) degree d ($d \geq 2$),

$$f(x, y, z) = \sum_{(i,j,k) \in \mathcal{J}} c_{i,j,k} \, N_{i,d}(x) \, N_{j,d}(y) \, N_{k,d}(z) \tag{1}$$

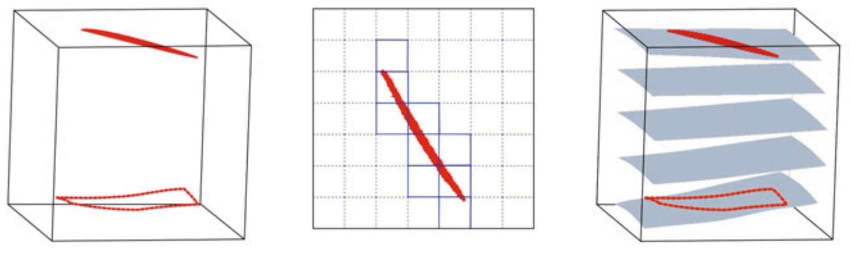


Fig. 2. From left to right: The blade boundary and the tip. The tensor-product splines are defined by three uniform knot sequences (top view). Several level sets $\mathcal{F}_\alpha$ of the scalar field.

with the real coefficients (control points) $c_{i,j,k}$, where $\mathcal{J}$ is the appropriate index set. The basis functions $(N_{i,d}(x)_{i=1,\ldots,n_i})$, $(N_{j,d}(y)_{j=1,\ldots,n_j})$ and $(N_{k,d}(z)_{k=1,\ldots,n_k})$ are B-splines of degree d with respect to uniform knot sequences $\mathcal{X} = (\beta_i)_{i=1,\ldots,n_i+1}$, $\mathcal{Y} = (\eta_j)_{j=1,\ldots,n_j+1}$ and $\mathcal{Z} = (\zeta_k)_{k=1,\ldots,n_k+1}$ for the three coordinate directions x, y and z. These knot sequences partition a bounding box Ω (domain of the tensor-product spline functions) into axis-aligned boxes. By using B-splines as basis functions we can use their advantageous properties such as local support and the increased flexibility compared to polynomials.

3.1 Slicing Surfaces for the Airfoil Part

We construct a family of slicing surfaces $\mathcal{F}_\alpha$ between the endwall ($\alpha = 0$) and the tip ($\alpha = 1$) of the blade, so that they generate well-behaved intersections (single connected components) for the airfoil part of the blade. The surfaces are designed in three steps, which are visualized in Fig. 2:

1. The tip $\mathcal{T}$ is represented by all tip points of the mesh, automatically detected by an adapted region-growing algorithm, while the endwall is represented by the blade boundary $\mathcal{C}_B$.
2. We fit these two data sets simultaneously using the techniques described in [12] (by minimizing the squared algebraic distances of a function f_α, constrained by some additional normal- and tension-terms) and by generalizing the term of minimizing the squared algebraic distances to

$$\sum_{\mathbf{v}\in\mathcal{C}_B} [f(\mathbf{v})]^2 + \sum_{\mathbf{v}\in\mathcal{T}} [f(\mathbf{v}) - 1]^2 . \tag{2}$$

   The B-spline basis functions are defined on the bounding box with uniform knot sequences for all three dimensions.
3. Finally, a finite subset of the set of level sets of the scalar field f defines the airfoil slicing surfaces

$$\mathcal{F}_\alpha = \{(x, y, z) \in \Omega \mid f_\alpha(x, y, z) = \alpha\} \quad \text{for } \alpha \in [\bar{\alpha}, 1] , \tag{3}$$

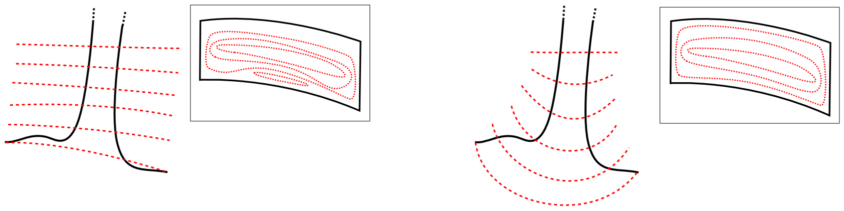


Fig. 3. The intersections of the slicing surfaces and the mesh have to be well-behaved and uniformly distributed in sweeping direction. To meet these conditions, we bend them up from horizontal to vertical positions.

where the value $\bar{\alpha}$ is chosen such that $\mathcal{F}_{\bar{\alpha}}$ $(0 < \bar{\alpha} < 1)$ is the lowest surface intersecting the airfoil part.

3.2 Slicing Surfaces for the Base Part

The family of (airfoil) slicing surfaces $\mathcal{F}_\alpha$ cannot be used for the base part because their intersections with the blade base are not well-behaved, see Fig. 3. Hence we define a second family of slicing surfaces $\mathcal{F}_\beta$ for the base part which we have to bend up from a horizontal position at the airfoil transition to a vertical position at the blade boundary.

In general, the transition curves between endwall and fillet are not useful to slice the base part, because there exists no clear segmentation between them in modern turbine blade geometries. Usually the fillet (and the endwall) is truncated at the front and rear of the endwall to save space and the endwall is designed with streamlined elements transitioned into the fillet, e.g. with a non-axisymmetric endwall contouring [8]. To address this problem we reconstruct the *fillet curve* $\mathcal{C}_F$ as a clear and significant feature for the base part of the blade.

The *fillet curve* $\mathcal{C}_F$ is the intersection of the airfoil and the endwall *before* the fillet has been generated. As a consequence, this curve $\mathcal{C}_F$ is a valuable indicator for the natural flow of the blade and for the transition between the endwall and the airfoil. Fig. 4 illustrates the reconstruction of the fillet curve.

Now, with the help of the fillet curve, we can define a family of ruled surfaces obtained by auxiliary slices and directions vectors. We define the *auxiliary slices* in four steps, see Fig. 5:

1. An algebraic spline surface is fitted to the airfoil data. The intersections of this surface with a family of airfoil slicing surfaces $\mathcal{F}_\alpha$ for $0 < \alpha < \bar{\alpha}$ defines the auxiliary airfoil slices.
2. The blade boundary curve $\mathcal{C}_B$ and the fillet curve $\mathcal{C}_F$ are projected on a cone whose axis equals the turbine axis.
3. A scalar field f is fitted to the curves $\mathcal{C}_B$ and $\mathcal{C}_F$ forced to $f = 0$ for $\mathcal{C}_B$ and $f = 1$ for $\mathcal{C}_F$. We use the same techniques as for the scalar field of the airfoil slicing surfaces.

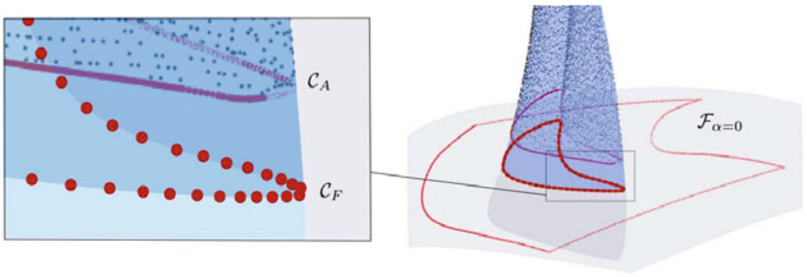


Fig. 4. To reconstruct the fillet curve $\mathcal{C}_F$, we fit an algebraic spline surface to all airfoil data points above the curve $\mathcal{C}_A$ and intersect this surface with the endwall surface $\mathcal{F}_{\alpha=0}$. $\mathcal{C}_A$ denotes the intersection of the lowest airfoil slicing surface $\mathcal{F}_{\alpha=\bar{\alpha}}$ with the blade.

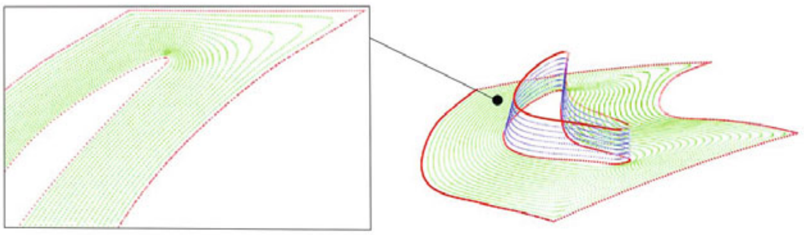


Fig. 5. The endwall surface and the (extended) airfoil surface are associated with fitted scalar fields. There level sets defines the auxiliary slices.

4. The level sets of the scalar field f defines the auxiliary endwall slices which we finally project back on the endwall surface $\mathcal{F}_{\alpha=0}$.

The corresponding *slicing directions* are defined in five steps, see Fig. 6:

1. The auxiliary endwall slices are divided into an inner and an outer part. The transition between them is called *transition slice* and their position is defined by the user.
2. A transition surface is defined as a ruled surface which is produced by the transition slice and directions which originate in the center of the turbine axis.
3. The intersection of this transition surface and the airfoil slicing surface $\mathcal{F}_{\alpha=\bar{\alpha}}$ is called the *bending curve*.
4. The points on the outer auxiliary endwall slices are associated with vertical direction vectors that originate in the center of the turbine axis.
5. The points on the inner auxiliary endwall slices and on the auxiliary airfoil slices are associated with direction vectors that point to the closest points on the bending curve.

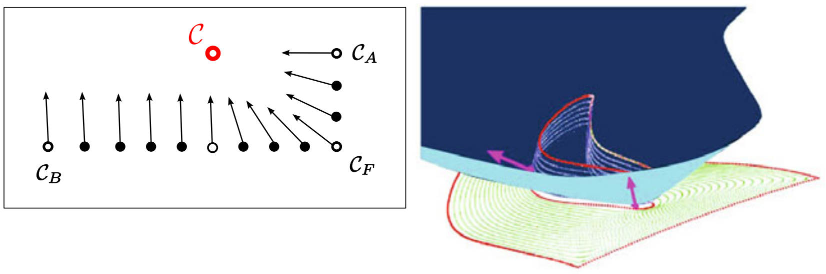


Fig. 6. Left: The slice directions are generated with the help of a bending curve $\mathcal{C}$. Right: A slicing surface.

Each of the auxiliary slices, along with the associated direction vectors, defines a ruled surface. We approximate these ruled surfaces by algebraic spline surfaces. This gives us the second family of slicing surfaces $\mathcal{F}_\beta$, see Fig. 6.

Remark. In our framework, the intersection curve of the two algebraic surfaces in step 3 is traced by a predictor-corrector method with curvature-based stepsize control, cf. [3,7]. This method generates a piecewise linear approximation of the bending curve.

4 Segmentation of the Slices

The (closed) piecewise linear slices $\mathcal{C}_\omega$ are generated by intersecting the mesh with the slicing surfaces $\mathcal{F}_\alpha$ and $\mathcal{F}_\beta$. In addition we have to consider two special cases: The lower blade boundary $\mathcal{C}_B$ can be taken directly from the mesh. For the uppermost slice, we intersect the slicing surface $\mathcal{F}_{\alpha=1}$ with an algebraic spline surface approximating the airfoil data. We cannot use the original mesh data here, since the mesh is slightly topped of and noisy on the transition between the tip and airfoil, due to errors introduced be the optical measurement process. Fig. 7 shows the resulting slices.

The slices $\mathcal{C}_\omega$ are now subdivided into four segments. This prepares them for the generation of the tensor-product spline volume. In addition, it complies with the standard turbine blade geometry, specially the classification of the airfoil into the edge and side parts.

4.1 Airfoil Part

In the airfoil part, the slices are called profiles (referring to the blade design process) and can be segmented by the wedge points into the two edge parts (trailing and leading edge) and the two side parts, see Fig. 8.

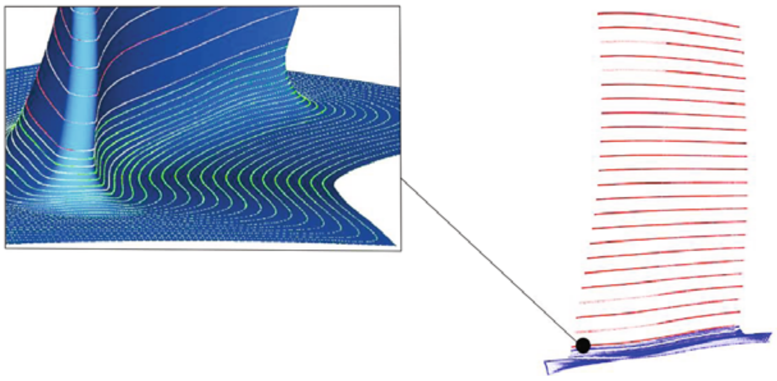


Fig. 7. Piecewise linear slices with (right) and without (left) the underlying mesh

Edge Points. In each profile, the two points where the extrapolated medial axis intersects the profile are called the *edge points*. We need to locate them, in order to perform a robust approximation of the (symmetric) wedge points later. Considering only one profile, the following heuristic method constructs an extended medial axis in the edge part of the profile and generates the corresponding edge point:

1. As an initial edge point $\mathbf{p}_{i=0}$ we take the point of the profile with the minimal (respectively maximal) x-coordinate. In general, these points are situated between their associated wedge points.
2. Starting from the edge point $\mathbf{p}_i$, we generate four auxiliary points on each side of the profile. Each pair of auxiliary points has a fixed distance $d, \ldots, 4d$ to the edge point, where d is a certain fraction of the profile length in x-direction.
3. These four points are used to estimate the
4. medial axis of the blade in this slice.
5. An new edge point $\mathbf{p}_{i+1}$ is found by collecting the point of the profile with the minimal distance to the extrapolated medial axis.
6. We continue with step 2 until the edge point has converged to a stable position.

Fig. 8 shows two steps of the edge point iteration.

Wedge Points. For a robust approximation of the wedge points, we have to consider the following requirements, which are due to the needs of our industrial application.

1. In general, the curvature is not a distinctive feature for a wedge point of a profile; less for the leading edge than for the trailing edge and less for turbine blades than for compressor blades.

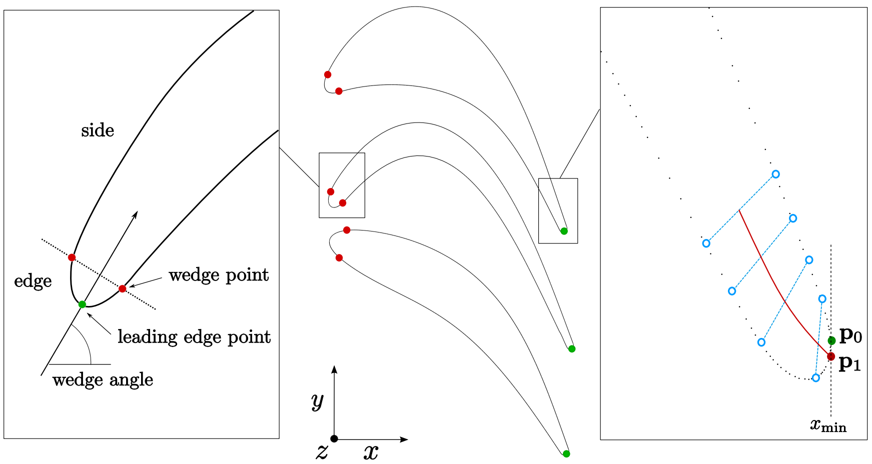


Fig. 8. From left to right: Some design parameters for the airfoil profiles. Three different types of profiles (two turbine blade profiles and one compressor blade profile at the bottom) with their wedge points for the leading edge and their trailing edge point for the trailing edge. Two steps of the edge point iteration process $\mathbf{p}_0 \rightarrow \mathbf{p}_1 \rightarrow \dots$ where a Bézier curve is generated by the center of four pairs of auxiliary points, giving a new approximation of the edge point.

2. The optical measurement process generates some remarkable data noise in regions of high curvature, e.g. in the trailing and leading edge. This leads to low data quality in the wedge point regions.
3. The consistency of the wedge points for all profiles is considerably more important than the exact detection of the wedge points for one profile.

As a consequence, we assume symmetric wedge points (in relation to the edge points) and approximate the edge parts of the profiles with spheres. More precisely, we start with the edge point, consider one neighboring curve point on each side of the profile and generate a sphere through these points, where we force the center of the sphere to be located in the plane spanned by the points. Then, we increase the number of (symmetric) neighboring profile points by a adding neighboring points on both ends and create fitting spheres (using the method described in [16]) with centers located in the plane of regression defined by them. This process of adding points is stopped when the approximation error starts to increase linearly with the number of points, i.e., when the profile segment starts to deviate substantially from a spherical curve. At this point, the sphere detaches itself from the edge part of the profile. This point defines the wedge points. In order to increase the consistency of the wedge points over all profiles we use standard data smoothing techniques. See Fig. 9 for some results.

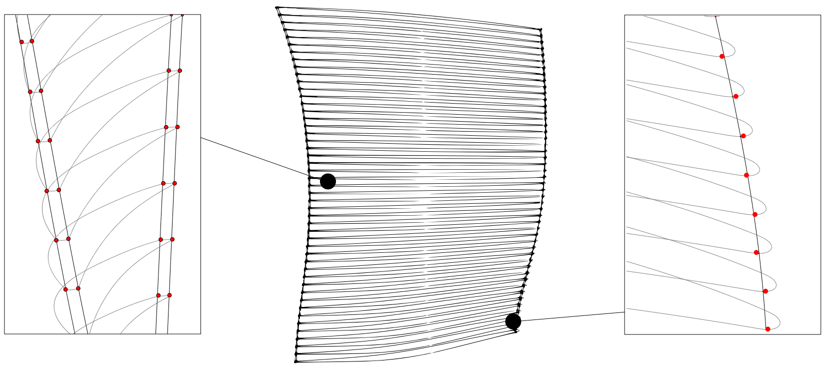


Fig. 9. Wedge points for all profiles. The plot on the right-hand side visualizes the smoothing process.

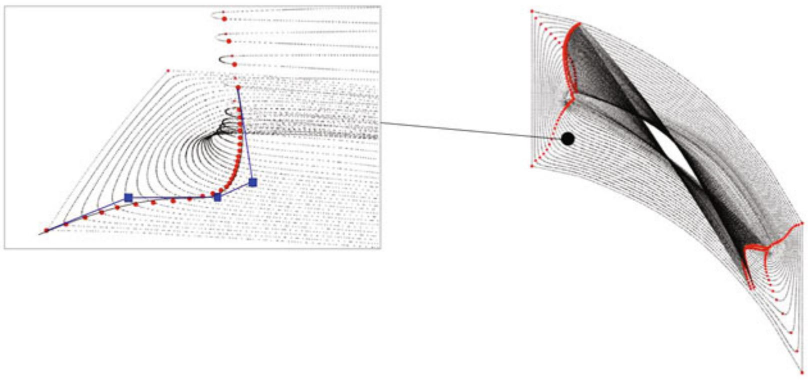


Fig. 10. Based on the tangential and a normal direction of the airfoil wedge curves and the bisector of the endwall surface at the corners, an auxiliary Bézier curve of degree four is defined. The slice points with minimal distances to this curve defines the wedge points for the base part.

4.2 Base Part

There exists no precise definition of edge points or wedge points for the slices of the base part. Therefore we extend the wedge points of the airfoil part smoothly into the base part, see Fig. 10.

5 Curve Fitting

In order to generate a volumetric tensor-product B-spline model, we have to represent the closed piecewise linear slices $\mathcal{C}_\omega$ by closed B-spline curves $\mathbf{c}_\omega(u)$

with identical degree and knot vector. We first generate initial B-spline curves and optimize them using an iterative fitting process.

5.1 Initial B-Spline Curves

For each slice $\mathcal{C} \in \mathcal{C}_\omega$, the initial B-spline curve $\mathbf{c}(u)$,

$$\mathbf{c}(u) = \sum_{i=0}^{n} N_{i,d}(u)\,\mathbf{d}_i, \tag{4}$$

with B-spline basis functions $N_{i,d}$ of degree d with respect to a periodic knot sequence $\mathcal{U}$ and control points $\mathbf{d}_i \in \mathbb{R}^3$, is generated in four steps:

1. We develop the piecewise linear slice $\mathcal{C}$ to $\mathcal{U} = [0,1]$ and relate the four wedge points to their parameters on $\mathcal{U}$, called wedge knots.
2. The remaining knots are distributed in the resulting four segments of $\mathcal{U}$, as specified by the wedge knots, by taking into account two requirements: First, the knots have to be distributed uniformly on each segment. Second, the length of the knot spans are identical for the edge parts and for the side parts.
3. We insert $d-2$ additional wedge knots to relax the built-in smoothness to C^1 because the transitions between edges and sides of a profile are designed to be C^1 only.
4. Finally, we calculate the Greville abscissas of the knot sequence $\mathcal{U}$ and use the related points on the slice $\mathcal{C}$ as the initial positions of the control points $\mathbf{d}_i$.

Considering the whole blade, all initial B-spline curves $\mathbf{c}_\omega(u)$ have to be based on identical knot sequences, in order to be able to extend them to a tensor-product B-spline volume. Therefore we perform the first three steps for only one slice (representing the average geometry of the profiles) and use the obtained knot vector for all slices.

5.2 Fitting Process

We consider a slice $\mathcal{C} \in \mathcal{C}_\omega$, which is described by the points $\mathbf{p}_j \in \mathbb{R}^3$ obtained by intersecting the initial triangulated data with one of the slicing surfaces. In addition, we have an initial B-spline curve $\mathbf{c}(u)$. The four wedge knots $\hat{u}_0, \hat{u}_1, \hat{u}_2, \hat{u}_3$ are linked to the four wedge points $\hat{\mathbf{p}}_0, \hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, \hat{\mathbf{p}}_3$.

In order to obtain a better approximation, we minimize the objective function

$$\sum_{k} \|\mathbf{p}_k - \mathbf{c}(u_k)\|^2 + \sum_{j=1}^{4} \|\hat{\mathbf{p}}_j - \mathbf{c}(\hat{u}_j)\|^2 \to \min_{\mathbf{d}_i, u_k}. \tag{5}$$

This problem is solved numerically by a Gauss-Newton-type method, which can be interpreted geometrically as an evolution process, as described in [1,13,15]. An overview about some standard B-spline fitting techniques are given in [6,17].

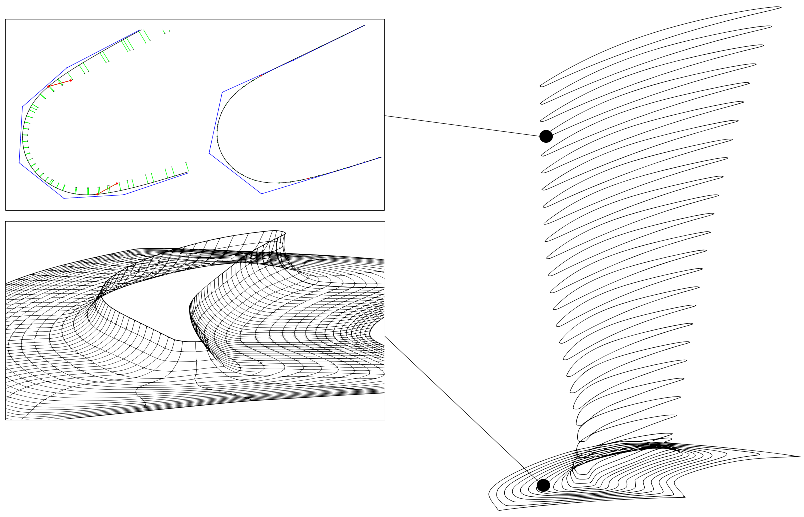


Fig. 11. Top left: One fitting step and the final B-spline curve of the evolution process. The red lines indicate the relation to the point data points with fixed parameters (wedge points, and possibly some additional points), and the green are associated with data points possessing floating parameters. Bottom left: A mesh obtained by piecewise linear interpolation of the B-splines curves. Right: The final set of B-spline curves that represent the blade surface.

In each iteration step, the data points $\mathbf{p}_j$ (with free parameters) are related to their closest points on the curve $\mathbf{c}(u_j)$ in a least-square sense. The wedge knots $\hat{u}_j$ are kept fixed. We use regularization terms (Tikhonov regularization combined with constraints on the tangential movement) in order to obtain a unique solution and a well-distributed parametric speed along the curve. Fig. 11 shows the resulting B-spline curves.

The profile curves are fitted independently, except for using identical degrees, knot sequences and wedge knots. This can lead to some unacceptable distortions of the curve parametrizations between the wedge knots by comparing neighboring curves. We suggest two approaches to overcome this problem. First, one may use more (uniformly distributed) points with fixed parameters between the wedge knots. Alternatively, one may apply a smoothing step (simultaneously across all profile curves) to the parametrization of the closest points of the data points after every iteration step. Fig. 11 shows the optimized curve parametrizations for the base part.

The used fitting framework is applicable to all manifolds of curves which are controlled by a certain system of shape parameters. Thus it is a highly valuable tool for turbine blade engineering. For example, in the aerodynamic optimization process the airfoil is designed by (lofted) streamlines represented as four C^1-connected Bézier curves of a fixed degree d, and the method can easily be adapted to this situation.

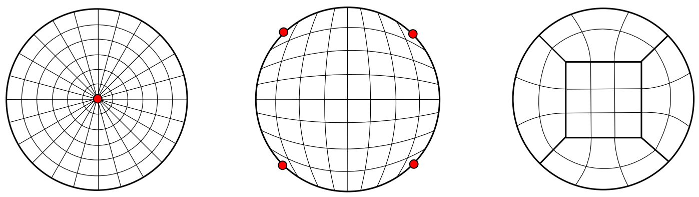


Fig. 12. Three possible parametrizations of a circular patch: A single-patch representation with a highly singular point in the center (left), a single-patch quadrilateral parametrization with four singular points on the boundary (center), and a regular five-patch representation (right)

6 Surface Generation

In this section we describe how to generate a B-spline surface for each slice. In the next section, we collect these slices to form a volume parametrization of the entire blade.

Any slice of the blade is topologically equivalent to a circle. A circular surface patch can be parametrized using different patch layouts, see Fig. 12 for three of them. In consideration of the blade proportions and the partition of the airfoil profiles by the wedge points, we use the quadrilateral representation with four singular points on the boundary which we relate to the wedge points. In the future, we plan to use the regular five-patch representation to generate a collection of tensor-product spline volumes ('multiple-patch volume') respecting the wedge point segmentation. Also, we plan to construct parametrizations that respect the interior structure of the blade, where cooling channels may be present.

The B-spline surfaces $\mathbf{s}_\omega(u, v)$ are generated by extending the B-spline curves $\mathbf{c}_\omega(u)$ in three steps, as follows:

1. The closed B-spline curve $\mathbf{c}(u)$ is subdivided at the wedge points by inserting one more knot on each wedge knot. This results in four B-spline curves of degree d and with one of the two different knot sequences $\mathcal{U}$ and $\mathcal{V}$.
2. The control points of the four curves are extended to the inner by a bilinearly blended discrete Coons patch $\mathbf{d}_{i,j}$, see [5].
3. For the base part, we move the inner control points smoothly in direction to the turbine axis to ensure a well-behaved (non-intersecting) parametrization there.

Finally, for each slice, we obtain a tensor-product B-spline surface

$$\mathbf{s}(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,d_u}(u)\ N_{j,d_v}(v)\ \mathbf{d}_{i,j}, \tag{6}$$

with degrees $d_u = d_v = d$, knot sequences $\mathcal{U}, \mathcal{V}$ and control points $\mathbf{d}_{i,j}$. Fig. 13 shows several of the generated B-splines surfaces $\mathbf{s}_\omega(u, v)$.

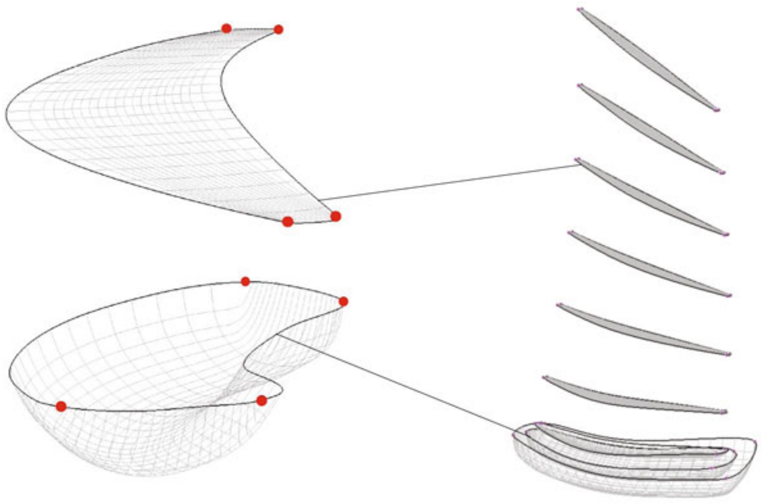


Fig. 13. Several B-spline surfaces representing the slices of the blade are shown. Two of them (left) are shown in detail. The highlighted surface points are the singularities of the patches, which correspond to the profile's wedge points.

7 Volume Generation

All B-spline surfaces $\mathbf{s}_\omega(u, v)$ have identical degrees and knot sequences. Thus we can generate a tensor-product B-spline volume $\mathbf{v}(u, v, w)$,

$$\mathbf{v}(u, v, w) = \sum_{i=0}^{n} \sum_{j=0}^{m} \sum_{k=0}^{l} N_{i,d_u}(u)\ N_{j,d_v}(v)\ N_{k,d_w}(w)\ \mathbf{d}_{i,j,k}, \tag{7}$$

by interpolating the surfaces in three steps:

1. The user defines a degree d_w.
2. A knot sequence $\mathcal{W}$ is defined by averaging the chord length parametrization of the surface points $\mathbf{s}_\omega(0.5, 0.5)$.
3. The control points $\mathbf{d}_{i,j,k}$ are generated by interpolating the control points of all surfaces with B-spline curves of degree d_w and knot sequence $\mathcal{W}$.

In order to cover the entire blade geometry, we add an appropriate trivariate B-spline block to the base part of the blade. Fig. 14 shows the final tensor-product B-spline volume.

8 Conclusions

The paper proposed a framework to model a single trivariate B-spline of a turbine blade from input triangle meshes of the blade. In order to be compatible with

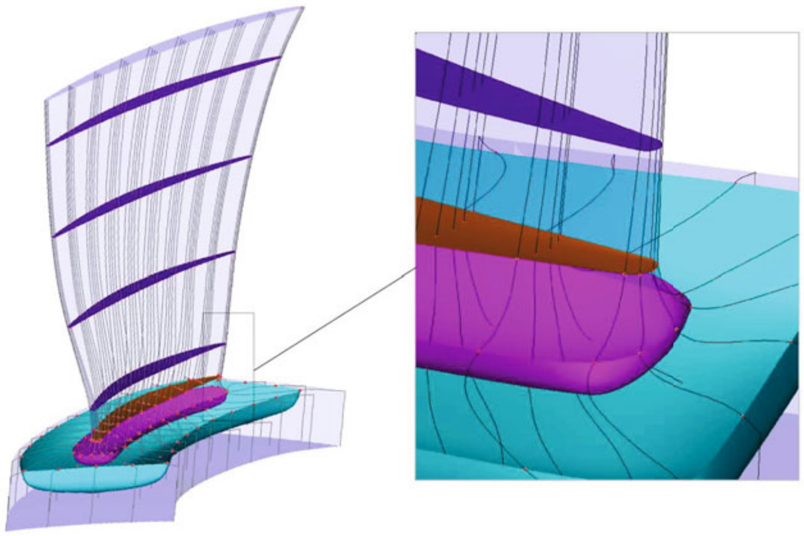


Fig. 14. The final trivariate B-spline model. The figure shows several parametric curves and surfaces.

the industrial process for blade design, we considered all standard blade features, such as wedge points, fillet curve and the natural flow of the blade at the fillet, when generating the parametrization.

The surface of the blade is represented by one surface patch with a smooth and well-behaved bivariate B-spline parametrization between the endwall boundary and the tip. It is ready for the (standard) commercial CAD and CAE applications. The volume is represented by a single trivariate B-spline parametrization and therefore usable both for isogeometric simulations and for a fully automatic structured mesh generation with standard finite elements.

To increase the quality of the model in the future, we want to investigate in the quality of the inner parametrization of the base part and in the generation of a multi-patch volumetric representation, respecting the blade features. Furthermore, local refinement techniques should be useful to avoid the restrictive tensor-product property of the B-splines geometries which may lead to distortions in the blade parametrization.

The framework is implemented in *Common LISP* and generates a volumetric blade model in a fully-automatic process using the user-defined degrees of freedom for the resulting model. For a triangle mesh with around 300.000 data points as input, the total computation time is around 20min, based on an Intel Core 2 duo processor with 3.0GHz and 8GByte memory. Nevertheless all steps of the process can be used separately in an object-oriented framework for blade modeling.

References

1. Aigner, M., Jüttler, B.: Approximation Flows in Shape Manifolds. In: Chenin, P., Lyche, T., Schumaker, L.L. (eds.) Curve and Surface Design: Avignon 2006, pp. 1–10. Nashboro Press (2007)
2. Aigner, M., Heinrich, C., Jüttler, B., Pilgerstorfer, E., Simeon, B., Vuong, A.-V.: Swept Volume Parameterization for Isogeometric Analysis. In: Hancock, E.R., Martin, R.R., Sabin, M.A. (eds.) Mathematics of Surfaces XIII. LNCS, vol. 5654, pp. 19–44. Springer, Heidelberg (2009)
3. Bajaj, C.L., Hoffmann, C.M., Lynch, R.E., Hopcroft, J.E.H.: Tracing surface intersections. Computer Aided Geometric Design 5, 285–307 (1988)
4. Boehm, W., Prautzsch, H.: Numerical Methods. A. K. Peters, Wellesley (1993)
5. Farin, G., Hansford, D.: Discrete Coons patches. Computer Aided Geometric Design 16, 691–700 (1999)
6. Floater, M.S.: Meshless parameterization and B-spline surface approximation. In: Cipolla, R., Martin, R. (eds.) The Mathematics of Surfaces IX, pp. 1–18. Springer, Heidelberg (2000)
7. Goldman, R.: Curvature Formulas for Implicit Curves and Surfaces. Computer Aided Geometric Design 22, 632–658 (2005)
8. Gregory-Smith, D.G., Ingramnd, G., Jayaraman, P., Harvey, N.W., Rose, M.G.: Non-axisymmetric turbine end wall profiling. Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy 215, 721–734 (2001)
9. Hoschek, J., Lasser, D.: Fundamentals of Computer Aided Geometric Design. A. K. Peters, Wellesley (1993)
10. Hughes, T.J.R., Cottrell, J.A., Bazilevs, Y.: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Computer Methods in Applied Mechanics and Engineering 194, 4135–4195 (2005)
11. Jüttler, B.: Least-squares fitting of algebraic spline curves via normal vector estimation. In: Cipolla, R., Martin, R. (eds.) The Mathematics of Surfaces IX, pp. 263–280. Springer, Heidelberg (2000)
12. Jüttler, B., Felis, A.: Least-squares fitting of algebraic spline surfaces. Advances in Computational Mathematics 17, 135–152 (2002)
13. Liu, Y., Wang, W.: A Revisit to Least Squares Orthogonal Distance Fitting of Parametric Curves and Surfaces. In: Chen, F., Jüttler, B. (eds.) GMP 2008. LNCS, vol. 4975, pp. 384–397. Springer, Heidelberg (2008)
14. Martin, T., Cohen, E., Kirby, R.M.: Volumetric parameterization and trivariate B-spline fitting using harmonic functions. Computer Aided Geometric Design 26, 648–664 (2009)
15. Song, X., Aigner, M., Chen, F., Jüttler, B.: Circular spline fitting using an evolution process. J. of Computational and Applied Mathematics 231, 423–433 (2009)
16. Taubin, R.: Estimation of Planar Curves, Surfaces, and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence 13, 1115–1138 (1991)
17. Wang, W., Pottmann, H., Liu, Y.: Fitting B-spline curves to point clouds by squared distance minimization. ACM Trans. Graphics 25, 214–238 (2006)
18. Xu, G., Bajaj, C.: Regularization of B-spline objects. Computer Aided Geometric Design 28, 38–49 (2011)

Globally Convergent Adaptive Normal Multi-scale Transforms

Stanislav Harizanov[⋆]

Jacobs University Bremen, School of Engineering and Science,
Campus Ring 1, 28759 Bremen, Germany
s.harizanov@jacobs-university.de

Abstract. In this paper we introduce a family of globally well-posed and convergent normal multi-scale transforms with high-order detail decay rate for smooth curves, based on adaptivity. For one of the members in the family, we propose a concrete algorithm what the adaptive criteria should be, and provide numerical evidence for the implementation. We compare the performance of our algorithm with other normal multi-scale transforms.

Keywords: Nonlinear multi-scale transforms, curve representation, linear subdivision, global convergence, well-posedness, stability, detail decay.

1 Introduction

Normal multi-scale transforms (MTs) for curves and surfaces were introduced in [6], and have been used for multi-scale representation and compression of geometric objects [9,10,5], for adaptive approximation of level curves [2], in image analysis [1], and recently for interface tracking [14]. They allow for the efficient computational processing of densily sampled (or analytically given) geometric objects of co-dimension one. E.g., given a curve $\mathcal{C}$ in $\mathbb{R}^2$, an initial sequence of vertices $\mathbf{v}^0 \subset \mathcal{C}$ (creating a polygonal line interpolating $\mathcal{C}$), and a univariate subdivision operator T, a normal MT produces denser vertex sets $\mathbf{v}^j \subset \mathcal{C}$ and polygonal lines associated with them according to

$$\mathbf{v}^j = T\mathbf{v}^{j-1} + d^j \mathbf{n}^j, \qquad j \in \mathbb{N}, \tag{1}$$

where $(d^j \mathbf{n}^j)_i = d_i^j \mathbf{n}_i^j,\ i \in \mathbb{Z}$. In addition to the linear prediction $T\mathbf{v}^{j-1}$ (a set of points in $\mathbb{R}^2$ not necessarily on $\mathcal{C}$), the recursion entails the prediction of a set of unit "normals" $\mathbf{n}^j$ (typically unit normal vectors with respect to the edges of the polygonal line associated with $\mathbf{v}^{j-1}$), and storing sequences of scalar "details" d^j which is enough for the reconstruction of $\mathbf{v}^j$ from $\mathbf{v}^{j-1}$. In the analysis step, a new point $\mathbf{v}^j = T\mathbf{v}^{j-1} + d^j \mathbf{n}^j$ resp. detail coefficient d_i^j requires

[⋆] The author is supported by the German Research Foundation (DFG) under grant OS-122/3-1.

the determination of intersection points of the "normal lines" $T\mathbf{v}_i^{j-1} + t\mathbf{n}_i^j$ with $\mathcal{C}$. Virtually the same approach, now with a bivariate subdivision operator T, works for surfaces in $\mathbb{R}^3$.

Due to the different nature of data and details, and the obvious nonlinearity of the normal MT

$$\mathcal{C} \quad \longleftrightarrow \quad \{\mathbf{v}^0, d^1, d^2, \ldots\},$$

given by (1), the mathematical analysis of normal MTs is nontrivial. Normal MTs for smooth curves in $\mathbb{R}^2$ have been investigated in [3,13,8] in quite some detail, while a first step towards a theory of normal MTs for smooth surfaces in $\mathbb{R}^3$ has been made only very recently in [12]. Most of the results in these papers are of asymptotic nature, i.e., they guarantee well-posedness, convergence, fast detail decay, etc. of normal MTs only for dense enough, regularly spaced initial point sets $\mathbf{v}^0$. In contrast, global well-posedness and convergence of normal MTs for any initial $\mathbf{v}^0 \subset \mathcal{C}$ can be expected only for schemes with very local T, such as the linear spline subdivision operator S_1 [3,13] or the Chaikin subdivision operator S_2 [8], which however comes at the price of undesirably slow detail decay for smooth curves. A natural remedy is the introduction of normal MTs based on the combination of a globally convergent scheme (used for small j) with schemes that guarantee better asymptotic detail decay.

The main goal of this paper is to propose and test a globally convergent adaptive scheme based on this idea. We consider a combination of normal MTs based on the Chaikin subdivision operator S_2 and the 4-point subdivision operator (also known as interpolating cubic Deslauriers-Dubuc scheme), in the following denoted by T. In Section 2 we introduce S_2, and recall basic properties of the associated normal MT established in [8]. In addition, we provide a stability result for this normal MT following [3,13]. In Section 3 we consider a new normal MT based on combining S_2 and T, and propose a concrete, locally adaptive algorithm. In Section 4, we apply this adaptive normal MT to simple test curves, and compare it to other normal MTs. Finally, we also compare our adaptive procedure with the one proposed in [3] based on the family of 4-point subdivision operators $T_\omega = (1-\omega)S_1 + \omega T$.

2 Chaikin Normal MT

Throughout this paper, we follow the usual simplifying assumption that univariate subdivision operators and schemes are defined on sequences indexed over $\mathbb{Z}$. Boldfaced letters are used for vector-valued quantities (points in $\mathbb{R}^2$, sequences of such points, or $\mathbb{R}^2$-valued functions). We work only with ℓ^∞ norms, i.e.,

$$\|\mathbf{v}\| = \sup_{i\in\mathbb{Z}} |\mathbf{v}_i|,$$

hence we will not explicitly indicate it. The notation $|\cdot|$ is used both for the absolute value in $\mathbb{R}$ and the Euclidean norm in $\mathbb{R}^2$ which should be clear from the context. We denote by Δ the difference operator $(\Delta\mathbf{v})_i = \mathbf{v}_{i+1} - \mathbf{v}_i,\ i \in \mathbb{Z}$. We assume that the initial curve $\mathcal{C}$ is closed, non-self-intersecting, and always given by its arc-length parameterization $\mathbf{v}(s) : [0, L] \to \mathbb{R}^2$, where L is the length of $\mathcal{C}$. To each point $\mathbf{v}_i^j \in \mathcal{C}$ we associate $s_i^j \in [0, L)$ such that $\mathbf{v}_i^j = \mathbf{v}(s_i^j)$. Whenever we write $\mathcal{C} \in C^{k,\alpha}$ we mean that $\mathbf{v}(s) \in C^{k,\alpha}$, i.e., $\mathbf{v}(s) \in C^k$ and the k-th derivative $\mathbf{v}^{(k)}(s)$ is Hölder continuous with exponent α.

The Chaikin subdivision operator S_2 is defined via

$$(S_2\mathbf{v})_{2i} = \frac{3\mathbf{v}_i + \mathbf{v}_{i+1}}{4}, \qquad (S_2\mathbf{v})_{2i+1} = \frac{\mathbf{v}_i + 3\mathbf{v}_{i+1}}{4}, \qquad i \in \mathbb{Z}. \tag{2}$$

S_2 yields a linear approximating subdivision scheme, also known in the literature as the *corner-cutting* or *the quadratic B-spline* scheme. Its Hölder smoothness exponent is $s_\infty(S_2) = 2$, meaning that S_2 generates $C^{1,1}$ functions as limit.

With the natural choice of normal directions

$$\mathbf{n}_{2i}^j = \mathbf{n}_{2i+1}^j \perp \Delta\mathbf{v}_i^{j-1}, \qquad i \in \mathbb{Z}, \quad j \geq 1, \tag{3}$$

and of points of intersection $\mathbf{v}_{2i}^j = \inf_{s > s_i^{j-1}}\{\mathbf{v}(s) : \mathbf{v}(s) \in L_{i,0}\}$ and $\mathbf{v}_{2i+1}^j = \sup_{s < s_{i+1}^{j-1}}\{\mathbf{v}(s) : \mathbf{v}(s) \in L_{i,1}\}$ we obtain the S_2 normal MT studied in [8]. Here $L_{i,\ell}(t) = (S_2\mathbf{v}^{j-1})_{2i+\ell} + t\mathbf{n}_{2i+\ell}^j$, $\ell = 0, 1$. Since we are only interested in the size of the details, the orientation of $\mathbf{n}_{2i}^j$ does not matter.

Theorem 1. *Let $\mathcal{C}$ be a closed, non-self-intersecting $C^{1,\alpha}$ curve, $0 < \alpha \leq 1$. For any (ordered) initial point sequence $\mathbf{v}^0 \subset \mathcal{C}$ the S_2 normal MT that satisfies (3) is well-defined. It produces point sequences $\mathbf{v}^j \subset \mathcal{C}$ and scalar detail sequences $d^j = (d_i^j)_{i\in\mathbb{Z}}$ such that*

$$\|\Delta\mathbf{v}^j\| \leq C_0\|\Delta\mathbf{v}^0\|2^{-j}, \quad \|\Delta^2\mathbf{v}^j\| \leq C_1\|\Delta\mathbf{v}^0\|2^{-j(1+\alpha')}, \quad j \geq 1, \tag{4}$$

$0 < \alpha' < \alpha$, *and*

$$\|d^j\| \leq C_2\|\Delta\mathbf{v}^{j-1}\|^{1+\alpha} \leq C_3(\|\Delta\mathbf{v}^0\|2^{-j})^{(1+\alpha)}, \qquad j \geq 1, \tag{5}$$

hold. The finite constants C_0, C_1, C_2, C_3 depend solely on the curve $\mathcal{C}$. Moreover, if $\{\tilde{\mathbf{v}}^0, \tilde{d}^1, \tilde{d}^2, \ldots, \tilde{d}^J\}$ with arbitrary $J \in \mathbb{N}$ is a perturbed representation of the actual multi-scale data, i.e.,

$$\|\mathbf{v}^0 - \tilde{\mathbf{v}}^0\| \leq \epsilon_0, \qquad \|d^j - \tilde{d}^j\| \leq \epsilon_d 2^{-j\nu}, \quad 1 \leq j \leq J,\ \nu > 0$$

then there exists a constant C_4, depending on $\mathcal{C}, \mathbf{v}^0, \epsilon_0$ and ϵ_d, but not on J, s.t.,

$$\|\mathbf{v}^j - \tilde{\mathbf{v}}^j\| \leq C_4(\epsilon_0 + \epsilon_d), \qquad \forall j \leq J. \tag{6}$$

The first part of Theorem 1 is exactly [8, Theorem 4.1]. The stability result (6) follows straightforwardly from (4), (5) and the proof of [13, Theorem 4], which for the sake of completeness we will repeat here. For any $j \in \mathbb{N}$, any $i \in \mathbb{Z}$, and $l \in \{0, 1\}$

$$\begin{aligned}|\mathbf{v}_{2i+l}^j - \tilde{\mathbf{v}}_{2i+l}^j| &\leq |S_2(\mathbf{v}^{j-1} - \tilde{\mathbf{v}}^{j-1})_{2i+l}| + |d_{2i+l}^j\mathbf{n}_{2i+l}^j - \tilde{d}_{2i+l}^j\tilde{\mathbf{n}}_{2i+l}^j| \\ &\leq \|\mathbf{v}^{j-1} - \tilde{\mathbf{v}}^{j-1}\| + \|d^j - \tilde{d}^j\| + |d_{2i+l}^j||\mathbf{n}_{2i+l}^j - \tilde{\mathbf{n}}_{2i+l}^j|.\end{aligned}$$

Now the main trick is to estimate the last term such that the perturbed data plays as less a role as possible, namely to verify

$$|\mathbf{n}^j_{2i+l} - \tilde{\mathbf{n}}^j_{2i+l}| \leq \frac{4\|\mathbf{v}^{j-1} - \tilde{\mathbf{v}}^{j-1}\|}{|\Delta \mathbf{v}^{j-1}_i|}. \tag{7}$$

Indeed, (3) gives rise to $\mathbf{n}^j_{2i+l} = (\Delta \mathbf{v}^{j-1}_i)^\perp / |\Delta \mathbf{v}^{j-1}_i|$, $\tilde{\mathbf{n}}^j_{2i+l} = (\Delta \tilde{\mathbf{v}}^{j-1}_i)^\perp / |\Delta \tilde{\mathbf{v}}^{j-1}_i|$, for both $l = 0, 1$. First, assume that $|\Delta \tilde{\mathbf{v}}^{j-1}_i| > 0$ and the $\mathbb{R}^2$ scalar product $\langle \Delta \mathbf{v}^{j-1}_i, \Delta \tilde{\mathbf{v}}^{j-1}_i \rangle \geq 0$. Then $|\mathbf{n}^j_{2i+l} - \tilde{\mathbf{n}}^j_{2i+l}| \leq 2\sqrt{2}\|\mathbf{v}^{j-1} - \tilde{\mathbf{v}}^{j-1}\| / |\Delta \mathbf{v}^{j-1}_i|$ holds. If $\langle \Delta \mathbf{v}^{j-1}_i, \Delta \tilde{\mathbf{v}}^{j-1}_i \rangle < 0$, then $2\|\mathbf{v}^{j-1} - \tilde{\mathbf{v}}^{j-1}\| > |\Delta \mathbf{v}^{j-1}_i|$ and (7) is fulfilled. Finally, unless some additional restrictions on ϵ_0, ϵ_d and ν are imposed, $|\Delta \tilde{\mathbf{v}}^{j-1}_i|$ can be zero and, thus, $\tilde{\mathbf{n}}^j_{2i+l}$ may not be defined. In this case take arbitrary unit vectors $\tilde{\mathbf{n}}^j_{2i}, \tilde{\mathbf{n}}^j_{2i+1}$ and let $\tilde{\mathbf{v}}^j_{2i+l} = \tilde{\mathbf{v}}^{j-1}_i + \tilde{d}^j_{2i+l}\tilde{\mathbf{n}}^j_{2i+l}$, $l = 0, 1$. From triangle inequality we have $2\|\mathbf{v}^{j-1} - \tilde{\mathbf{v}}^{j-1}\| \geq |\Delta \mathbf{v}^{j-1}_i|$, so (7) remains true.

Once (7) is established, we use the local version of (5) (see [8, Section 4.1] or [13, Theorem 2]). Namely, for a $C^{1,\alpha}$ curve $\mathcal{C}$ there exists a global constant C independent on j, i such that

$$|d^j_{2i+l}| \leq C|\Delta \mathbf{v}^{j-1}_i|^{1+\alpha}, \qquad l = 0, 1.$$

Using this, together with the first part of (4), we conclude that

$$\|\mathbf{v}^j - \tilde{\mathbf{v}}^j\| \leq (1 + \underbrace{2^{2-\alpha} C C_0^\alpha \|\Delta \mathbf{v}^0\|^\alpha}_{C} 2^{-j\alpha})\|\mathbf{v}^{j-1} - \tilde{\mathbf{v}}^{j-1}\| + \epsilon_d 2^{-j\nu}.$$

The latter gives rise to (6) (see [13] for more details).

Some remarks are in order: Even though (6) measures the L_∞ distance between the actual curve $\mathcal{C}$ and the perturbed limit $\tilde{\mathcal{C}}$, both given analytically by their normal re-parameterizations, the same bound is valid for the Hausdorff distance $dist_{\mathcal{H}}(\mathcal{C}, \tilde{\mathcal{C}})$ between the geometric curves. Indeed, at each level $j \in \mathbb{N}$ the Hausdorff distance between the piecewise linear interpolants of $\mathbf{v}^j$ and $\tilde{\mathbf{v}}^j$ does not exceed $\|\mathbf{v}^j - \tilde{\mathbf{v}}^j\|_\infty$, since the Hausdorff distance between two line segments is less or equal to the maximum of the distances between the corresponding end points of the segments. Under the assumptions of Theorem 1, (6) uniformly bounds the point-wise distance between the original data and the perturbed one (confirming that compression is a numerically stable procedure [13, Remark 1]) but none of the nice properties of $\mathcal{C}$ and $\mathbf{v}^j$ are automatically inherited by $\tilde{\mathcal{C}}$ and $\tilde{\mathbf{v}}^j$. Indeed, $\tilde{\mathbf{v}}^j$ may not be even well-defined (e.g., two neighboring vertices may coincide) or when it is well-defined, inequalities such as (4) do not necessarily hold. Furthermore, $\tilde{\mathcal{C}}$ may have an arbitrary number of self-intersection-points and is only continuous in general. However, it seems that small/controlled perturbations may still preserve the additional structure of the data, but to prove this, different approaches than the one presented in this section should be used (for example the one proposed in [7, Section 4.1]).

3 Globally Convergent Normal MTs Based on Adaptivity

3.1 Theoretical Approach

Comparing the S_2 normal MT with another non-adaptive globally convergent normal MT - the one based on the mid-point interpolating scheme S_1

$$(S_1\mathbf{v})_{2i} = \mathbf{v}_i, \qquad (S_1\mathbf{v})_{2i+1} = \frac{\mathbf{v}_i + \mathbf{v}_{i+1}}{2}, \qquad i \in \mathbb{Z},$$

studied in [3,13], one sees that the detail decay rate is the same. The aim of this section is to show that S_2 normal MT has the property to "uniformize" the fine-scale data $\mathbf{v}^j$. Hence, it is suitable for adaptive procedures together with any high-regular subdivision scheme T. In other words, using S_2 for prediction on finitely many coarse levels (the exact number depends on the initial data $\mathbf{v}^0$, the initial curve $\mathcal{C}$, and the choice of T) guarantees that if we switch to T afterwards, the normal MT will be well-defined and the detail decay rate will be as high as the smoothness of $\mathcal{C}$ and the regularity of T allow. Thus, storing more data in the beginning (S_2 is an approximating scheme, so the S_2 normal MT produces twice as many details as the S_1 normal MT) may potentially lead to a better compression later. This does not hold for S_1 normal MT. For example, let $\mathcal{C}$ be just a line segment and

$$|\Delta\mathbf{v}^0_{-1}| = 9|\Delta\mathbf{v}^0_0| = 9|\Delta\mathbf{v}^0_1|.$$

Let T be the 4-point (also known as Dubuc-Deslauriers) subdivision scheme

$$(T\mathbf{v})_{2i} = \mathbf{v}_i, \qquad (T\mathbf{v})_{2i+1} = \frac{-\mathbf{v}_{i-1} + 9\mathbf{v}_i + 9\mathbf{v}_{i+1} - \mathbf{v}_{i+2}}{16}, \qquad i \in \mathbb{Z}. \tag{8}$$

Then T is not well-defined on $\mathbf{v}^0$ since $(T\mathbf{v}^0)_1 = \mathbf{v}^0_1 = (T\mathbf{v}^0)_2$. Moreover, for each $j \geq 1$

$$|(\Delta S_1^j\mathbf{v}^0)_{-1}| = 9|(\Delta S_1^j\mathbf{v}^0)_0| = 9|(\Delta S_1^j\mathbf{v}^0)_1|,$$

so one can never switch to T around $\mathbf{v}^0_0$.

Theorem 2.5 in [8] states that if $\mathcal{C} \in C^{k,\alpha}$, $\delta \in (0,\alpha)$, and a (convergent) subdivision scheme T with $s_\infty(T) = m + \beta$ are given, there exist $C, h > 0$ such that for any initial data $\mathbf{v}^0$, satisfying

$$\|\Delta^2\mathbf{v}^0\| \leq C\|\Delta\mathbf{v}^0\|^{1+\delta} \leq Ch^{1+\delta}, \tag{9}$$

the T normal MT is well-defined. The detail decay rate is

$$\|d^j\|_\infty = \mathrm{O}(\|\Delta\mathbf{v}^0\|2^{-j\mu}), \qquad \mu < \min\{k+\alpha, m+\beta+1, P_e\},\ j \geq 1,$$

where P_e is the order of exact polynomial reproduction for T, introduced in [8]. C and h depend on $\mathcal{C}$, T and δ, so to show that S_2 always works in adaptive algorithms, we need the following

Proposition 1. *Let $\mathcal{C}$ be a closed non-self-intersecting $C^{k,\alpha}$ curve. Then, for any $\delta \in (0, \alpha)$ and any initial set $\mathbf{v}^0 \subset \mathcal{C}$*

$$\frac{\|\Delta^2 \mathbf{v}^j\|}{\|\Delta \mathbf{v}^j\|^{1+\delta}} \to 0, \qquad j \to \infty, \tag{10}$$

where $\mathbf{v}^j$ is the multi-scale data at level j, obtained from $\mathbf{v}^0$ via S_2 normal MT.

Proof. From (2) it follows that for any given sequence $\mathbf{v}^{j-1} \subset \mathcal{C}$,

$$(\Delta^2 S_2 \mathbf{v}^{j-1})_{2i} = (\Delta^2 S_2 \mathbf{v}^{j-1})_{2i+1} = \frac{\Delta^2 \mathbf{v}_i^{j-1}}{4}, \qquad \forall i \in \mathbb{Z}.$$

Therefore, $\|\Delta^2 S_2 \mathbf{v}^{j-1}\| = \|\Delta^2 \mathbf{v}^{j-1}\|/4$. (3) implies that $\mathbf{v}_{2i}^j$ and $\mathbf{v}_{2i+1}^j$ lie on parallel lines, orthogonal to $\Delta \mathbf{v}_i$. Hence, $|\Delta \mathbf{v}_{2i}^j|$ cannot be smaller than the distance $|(\Delta S_2 \mathbf{v}^{j-1})_{2i}| = |\Delta \mathbf{v}_i^{j-1}|/2$ between the lines. This gives rise to

$$\|\Delta \mathbf{v}^j\| \geq \frac{\|\Delta \mathbf{v}^{j-1}\|}{2}.$$

Now, denote by $r_j := \|\Delta^2 \mathbf{v}^j\|/\|\Delta \mathbf{v}^j\|^{1+\delta}$. Using (4), (5), and the above we derive

$$r_j \leq \frac{\|\Delta^2 \mathbf{v}^{j-1}\|/4 + 4\|d^j\|}{(\|\Delta \mathbf{v}^{j-1}\|/2)^{1+\delta}} \leq 2^{\delta-1} r_{j-1} + C 2^{-(j-1)(\alpha-\delta)} \leq q r_{j-1} + C q^{j-1},$$

where the constant C depends on C_0, C_2, and $\|\Delta \mathbf{v}^0\|$, while $q := 2^{(\delta-\alpha)} < 1$. Thus,

$$r_j \leq q^j r_0 + C j q^{j-1} \to 0, \qquad j \to \infty.$$

Proposition 1 implies that, for any given closed non-self-intersecting $C^{k,\alpha}$ curve $\mathcal{C}$ with $k + \alpha > 1$, any choice of a linear subdivision scheme T, and any choice of (at least two) initial points $\mathbf{v}^0 \subset \mathcal{C}$, there exists $j \in \mathbb{N}$ (that depends on all $\mathcal{C}$, T and $\mathbf{v}^0$), such that the T normal MT for $\mathcal{C}$ with initial data $\mathbf{v}^j$, obtained from $\mathbf{v}^0$ after j refinement steps based on the S_2 normal MT, is well-defined, converges and possesses the detail decay rate as predicted in [3, Theorem 6.3] and [8, Theorem 2.5].

3.2 Adaptive Algorithm

The direct implementation of the adaptive normal MT, introduced in Section 3.1 is wasteful, as we did not take into account the locality of the prediction operators. A reduction of the number of segments, where S_2 is used is desirable for faster detail decay and less data storage. The adaptive algorithm we propose here allows different prediction operators to be used for different neighborhoods within the same scale. From now on, unless something else is specified, T will denote the Dubuc-Deslauriers operator (8). Suppose a closed curve $\mathcal{C}$ with initial $\mathbf{v}^0 \subset \mathcal{C}$ is given, a constant $RB > 0$ has been (manually) chosen, and the normal

MT $\mathbf{v}^1, \mathbf{v}^2, \dots, \mathbf{v}^{j-1}$ up to level $j-1$ has been carried over. Any vertex $\mathbf{v}_i^{j-1}$ is marked by an extra bit, which is zero if it is an S *vertex*, i.e., predicted by S_2 or S_1, and one if it is a T *vertex*, i.e., predicted by T. Initially, in $\mathbf{v}^0$ all vertices are declared S vertices. Note that we work only with finite initial data $\mathbf{v}^0$, so $\mathbf{v}^{j-1}$ is finite, as well. Let its cardinality be N. The algorithm consists of the following steps

– Enlarge $\mathbf{v}^{j-1}$ in a cyclic way by adding two elements $\mathbf{v}_{-1}^{j-1} = \mathbf{v}_{N-1}^{j-1}, \mathbf{v}_0^{j-1} = \mathbf{v}_N^{j-1}$ to the left and three elements $\mathbf{v}_{N+1}^{j-1} = \mathbf{v}_1^{j-1}, \mathbf{v}_{N+2}^{j-1} = \mathbf{v}_2^{j-1}, \mathbf{v}_{N+3}^{j-1} = \mathbf{v}_3^{j-1}$ to the right.
– Create an empty set V^j and whenever a vertex is marked, add it to V^j.
– For each $i \in [1, N]$:
  - If one of the vertices $\mathbf{v}_i^{j-1}$ and $\mathbf{v}_{i+1}^{j-1}$ is an S vertex, while the other is a T vertex, use S_1 to predict a new node between $\mathbf{v}_i^{j-1}$ and $\mathbf{v}_{i+1}^{j-1}$. Compute the new vertex $\hat{\mathbf{v}}_i$. In case of multiple intersection points, take an arbitrary one among them. Store the corresponding detail, and mark $\hat{\mathbf{v}}_i$ as an S vertex. Do not mark $\mathbf{v}_i^{j-1}$ or $\mathbf{v}_{i+1}^{j-1}$.
  - Else, i.e., when $\mathbf{v}_i^{j-1}$ and $\mathbf{v}_{i+1}^{j-1}$ are both either S or T vertices, determine the local neighborhood $\bar{\mathbf{v}} = \{\bar{\mathbf{v}}_{i-1}, \bar{\mathbf{v}}_i, \bar{\mathbf{v}}_{i+1}, \bar{\mathbf{v}}_{i+2}\}$. $\bar{\mathbf{v}}$ may differ from $\mathbf{v}^{j-1}\big|_{[i-1,i+2]}$ and its exact derivation will be explained later. If

$$|\Delta^2\bar{\mathbf{v}}_{i-1}| \le RB\min\{|\Delta\bar{\mathbf{v}}_{i-1}|, |\Delta\bar{\mathbf{v}}_i|\},\ |\Delta^2\bar{\mathbf{v}}_i| \le RB\min\{|\Delta\bar{\mathbf{v}}_i|, |\Delta\bar{\mathbf{v}}_{i+1}|\}, \tag{11}$$

    holds and the normal line through $T\bar{\mathbf{v}}$ intersects the corresponding arc $\widehat{\mathbf{v}_i^{j-1}\mathbf{v}_{i+1}^{j-1}} = \{\mathbf{v}(s) : s_i^{j-1} < s < s_{i+1}^{j-1}\}$ at least once, use T as prediction operator. Store the detail for the new vertex $\hat{\mathbf{v}}_i$ (in case of multiple intersection points, take an arbitrary one among them) and mark all $\{\mathbf{v}_i^{j-1}, \hat{\mathbf{v}}_i, \mathbf{v}_{i+1}^{j-1}\}$ as T vertices. Else, use S_2 as prediction operator, compute two new vertices, mark them as S vertices and store the details. Do not mark $\mathbf{v}_i^{j-1}$ or $\mathbf{v}_{i+1}^{j-1}$.
– Go one more time through V^j, store a single copy for each of the vertices in it (those from $\mathbf{v}^{j-1}$ may have two copies at the beginning) and delete all the S vertices, surrounded by both T neighbors. What remains is $\mathbf{v}^j$.

Let us explain first how to construct $\bar{\mathbf{v}}$. T is a primal scheme, i.e., node-oriented (averages the values in the nodes of the grid), while S_2 is a dual one, i.e., interval-oriented (averages the mean values of the function between adjacent nodes of the grid). Hence, taking $\bar{\mathbf{v}} = \mathbf{v}_{i+1}^{j-1}$ is not always appropriate, because the transition from one to the other leads to highly irregular data on the next level. We create $\bar{\mathbf{v}}$ of a pure S or pure T type via $\bar{\mathbf{v}}_i = \mathbf{v}_i^{j-1}$, $\bar{\mathbf{v}}_{i+1} = \mathbf{v}_{i+1}^{j-1}$,

$$\bar{\mathbf{v}}_{i-1} = \begin{cases} \mathbf{v}_{i-1}^{j-1}, & \text{if } \mathbf{v}_{i-1}^{j-1} \text{ and } \mathbf{v}_{i-2}^{j-1} \text{ are of the same type,} \\ \frac{\mathbf{v}_{i-2}^{j-1}+\mathbf{v}_{i-1}^{j-1}}{2}, & \text{otherwise,} \end{cases}$$

and analogously for $\bar{\mathbf{v}}_{i+2}$. Note that our algorithm guarantees that $\mathbf{v}_{i-1}^{j-1}$ and $\mathbf{v}_{i-2}^{j-1}$ are of the same type whenever $\mathbf{v}_{i-1}^{j-1}$ and $\mathbf{v}_i^{j-1}$ are not, so we never average vertices of different nature! Taking equally spaced points on a line and applying the above procedure, randomly choosing for each i whether T or S_2 normal MT should be used, gives rise to neighborhoods $\bar{\mathbf{v}}$ that consist of equidistant points, again. On the other hand, the experimental results indicate that our algorithm quickly improves the regularity of an initially irregular data set.

Let us now explain the rationale behind (11) in more detail. In order to use as few S_2 and S_1 steps as possible, we need a very local criteria, such as (11), for the choice of the prediction operator. Note that (11) implies (9) with $\delta = 0$. There are several reasons why we chose to work with $\delta = 0$. First of all, the size of δ does not play any role for the detail decay rate, which can be seen from the proofs in [3,8]. Secondly, by relaxing the restriction (9) we favor the T normal MT, and, thus, avoid the S_2 normal MT as much as possible (of course, we do not have theoretical guarantee that the T normal MT is well-defined and we need to check it every time we want to use it). Finally, (11) allows us to relate our work to [3], since the quantity we bound is asymptotically the same as their non-uniformity measure

$$\mathcal{N}(s) := \sup_i \mathcal{N}(s_i) = \sup_i \max\{\Delta s_{i+1}/\Delta s_i, \Delta s_i/\Delta s_{i+1}\}. \tag{12}$$

Indeed, since $\mathbf{v}(s) \in C^{1,\alpha}$, there exists a constant $C < \infty$ such that

$$|\mathbf{v}'(s) - \mathbf{v}'(s')| \le C|s - s'|^\alpha, \qquad \forall s, s' \in [0, L).$$

Hence, by Taylor formula

$$\left| \frac{|\Delta^2 \mathbf{v}^j_{i-1}|}{|\Delta \mathbf{v}^j_i|} - \frac{|\Delta^2 s^j_{i-1}|}{\Delta s^j_i} \right| \le \frac{\left| \Delta s^j_i |\Delta \mathbf{v}^j_i - \Delta \mathbf{v}^j_{i-1}| - |\Delta \mathbf{v}^j_i||\Delta s^j_i - \Delta s^j_{i-1}| \right|}{(\Delta s^j_i)^2 - C(\Delta s^j_i)^{2+\alpha}}.$$

For the expression $|A|$ in the numerator we derive

$$\begin{aligned} A &\le C(\Delta s^j_i)^{2+\alpha} \left(1 + \left(\frac{\Delta s^j_{i-1}}{\Delta s^j_i} \right)^{1+\alpha} + \left| \frac{\Delta s^j_{i-1}}{\Delta s^j_i} - 1 \right| \right); \\ -A &\le \Delta s^j_i \underbrace{\left(|\Delta s^j_i - \Delta s^j_{i-1}| - |\Delta \mathbf{v}^j_i - \Delta \mathbf{v}^j_{i-1}| \right)}_{B}. \end{aligned}$$

To continue the estimations in the second line, we need to consider two cases.

$$\begin{aligned} 1) \quad & |\Delta s^j_i - \Delta s^j_{i-1}| \ge C((\Delta s^j_{i-1})^{1+\alpha} + (\Delta s^j_i)^{1+\alpha}) \implies \\ & |\Delta \mathbf{v}^j_i - \Delta \mathbf{v}^j_{i-1}| \ge |\Delta s^j_i - \Delta s^j_{i-1}| - C((\Delta s^j_{i-1})^{1+\alpha} + (\Delta s^j_i)^{1+\alpha}) \implies \\ & \qquad B \le C(\Delta s^j_i)^{1+\alpha} \left(1 + \left(\frac{\Delta s^j_{i-1}}{\Delta s^j_i} \right)^{1+\alpha} \right); \\ 2) \quad & |\Delta s^j_i - \Delta s^j_{i-1}| \le C((\Delta s^j_{i-1})^{1+\alpha} + (\Delta s^j_i)^{1+\alpha}) \implies \\ & \qquad B \le |\Delta s^j_i - \Delta s^j_{i-1}| \le C(\Delta s^j_i)^{1+\alpha} \left(1 + \left(\frac{\Delta s^j_{i-1}}{\Delta s^j_i} \right)^{1+\alpha} \right). \end{aligned}$$

Combining the above inequalities with the analogous ones for $\Delta\mathbf{v}_{i-1}^j$ resp. Δs_{i-1}^j, instead of $\Delta\mathbf{v}_i^j$ resp. Δs_i^j, and using $|x-1| \le |x|^{1+\alpha}+1$ we get

$$\left|\frac{|\Delta^2\mathbf{v}_{i-1}^j|}{\min\{|\Delta\mathbf{v}_{i-1}^j|, |\Delta\mathbf{v}_i^j|\}} - \frac{|\Delta^2 s_{i-1}^j|}{\min\{\Delta s_{i-1}^j, \Delta s_i^j\}}\right| \le \frac{2C\max\{\Delta s_{i-1}^j, \Delta s_i^j\}^\alpha(1+\mathcal{N}(s_{i-1}^j)^{1+\alpha})}{1-C\max\{\Delta s_{i-1}^j, \Delta s_i^j\}^\alpha}.$$

But $|\Delta^2 s_{i-1}^j|/\min\{\Delta s_{i-1}^j, \Delta s_i^j\} = \mathcal{N}(s_{i-1}^j)-1$, so

$$\sup_i\left|\frac{|\Delta^2\mathbf{v}_{i-1}^j|}{\min\{|\Delta\mathbf{v}_{i-1}^j|, |\Delta\mathbf{v}_i^j|\}} - (\mathcal{N}(s_{i-1}^j)-1)\right| \le \frac{2C\|\Delta s^j\|^\alpha(1+\mathcal{N}(s^j)^{1+\alpha})}{1-C\|\Delta s^j\|^\alpha}. \quad (13)$$

What we see from (13) is that, whenever $\|\Delta s^j\| \to 0$ and $\mathcal{N}(s^j) \to 1$, we have

$$\sup_i \frac{|\Delta^2\mathbf{v}_{i-1}^j|}{\min\{|\Delta\mathbf{v}_{i-1}^j|, |\Delta\mathbf{v}_i^j|\}} \to 0, \qquad j \to \infty. \quad (14)$$

Vice versa, due to

$$|s-s'| \ge |\mathbf{v}(s)-\mathbf{v}(s')| \ge q|s-s'|, \qquad \forall s, s' \in [0, L), \quad (15)$$

which holds for non-self-intersecting closed C^1 curves (see [13]), we have

$$\forall j \in \mathbb{N}\ \sup_i\left|\frac{|\Delta^2\mathbf{v}_{i-1}^j|}{\min\{|\Delta\mathbf{v}_{i-1}^j|, |\Delta\mathbf{v}_i^j|\}} - (\mathcal{N}(s_{i-1}^j)-1)\right| \le \frac{2Cq^\alpha\|\Delta\mathbf{v}^j\|^\alpha(1+q^{1+\alpha}\mathcal{N}(\mathbf{v}^j)^{1+\alpha})}{1-Cq^\alpha\|\Delta\mathbf{v}^j\|^\alpha},$$

and, thus, $\|\Delta\mathbf{v}^j\| \to 0$ together with (14) implies $\mathcal{N}(s^j) \to 1$, when $j \to \infty$.

Some remarks are in order. According to the Appendix A in [3], T is weakly contractive with bound $R = 3+2\sqrt{2}$, i.e., for any strictly increasing $u \in \ell_\infty(\mathbb{Z})$ such that $\mathcal{N}(u) \le 3+2\sqrt{2}$, Tu is also strictly increasing and $\mathcal{N}(Tu) \le \mathcal{N}(u)$. The bound is sharp and, together with (13), suggests the restriction $RB < 2+2\sqrt{2}$, because otherwise we allow applications of T normal MT even when $\mathcal{N}(s^j) > R$ which could even worsen the regularity of our data! Secondly, our algorithm can be generalized for smoother both interpolating schemes T and approximating schemes S. However, the transition between S and T vertices should always be smooth, i.e., only S_2 and the 4-point scheme can be applied to neighboring intervals. Then in each of the directions one can upgrade the prediction operator with a smoother one of the same type. For example, in order to work with the central interpolating scheme of degree 9, we intermediately have to use the central interpolating schemes of degree 3, 5 and 7, as well.

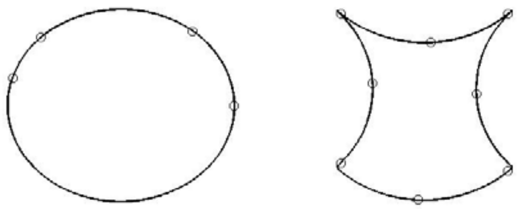


Fig. 1. Our test data

4 Experimental Results

For the two data sets illustrated in Fig. 1, we compare the behavior of the normal MTs based on S_1, S_2, T, S_1/T, and S_2/T (we use the notation S_2/T normal MT for the adaptive algorithm, presented in the previous section, and S_1/T normal MT for its exact counterpart, when S_2 predictions are replaced by S_1 predictions). In the first example, we consider a small, irregularly spaced initial vertex set $\mathbf{v}^0$ (indicated by circles) on a C^∞ curve, namely the unit circle. In the second example we consider well-spaced initial points $\mathbf{v}^0$ on a curve with four isolated singularities, which we will refer to as twisted circle, since it is obtained by dividing the unit circle into four arcs of equal length, and flipping three of them around their corresponding edges. Although close, none of the points in $\mathbf{v}^0$ coincides with any of the singularity points of the twisted circle. The twisted circle is only $C^{0,1}$. Hence the crucial inequality (15), needed for the well-posedness of the S_2 normal MT (see [8, (4.4)]), does not automatically hold. However, direct computations show that if P, Q lie on neighboring arcs of the twisted circle, then $|PQ| \geq |\widehat{PQ}|/(\pi\sqrt{2})$, which for the given choice of $\mathbf{v}^0$ is enough. As we will see, for both examples the T normal MT fails at some level.

In Fig. 2 we show experimental data for the first example. The non-uniformity measures $\mathcal{N}(\mathbf{v}^j)$ is computed in the same fashion as (12) by just replacing Δs_i by $|\Delta \mathbf{v}_i|$. We prefer to work with this measure, because at each level j it can be directly computed from $\mathbf{v}^{j-1}$, while for $\mathcal{N}(s^{j-1})$ one needs to take into account the underlying curve $\mathcal{C}$. In the second and third columns, by setting $RB = 0$, we simply apply pure S_2 resp. S_1 normal MT. The second row compares the detail decay, combining the details from all levels $j \leq 5$, while the last one compares the details only from the finest scale.

T normal MT fails after 6 iteration steps, and this can be observed by the rapidly increasing values of its non-uniformity measure. Since S_2, resp. S_1, normal MT is always well-defined, the adaptive transforms never fail and this is confirmed by the graphs of the associated non-uniformity measures. From these plots, we see the potential problem of working with large RB-values. Indeed, when we set $RB = 5$ (which is slightly larger than the theoretical bound $2+2\sqrt{2}$ for T, as discussed in Section 3.2), we allow T normal MT steps, that even

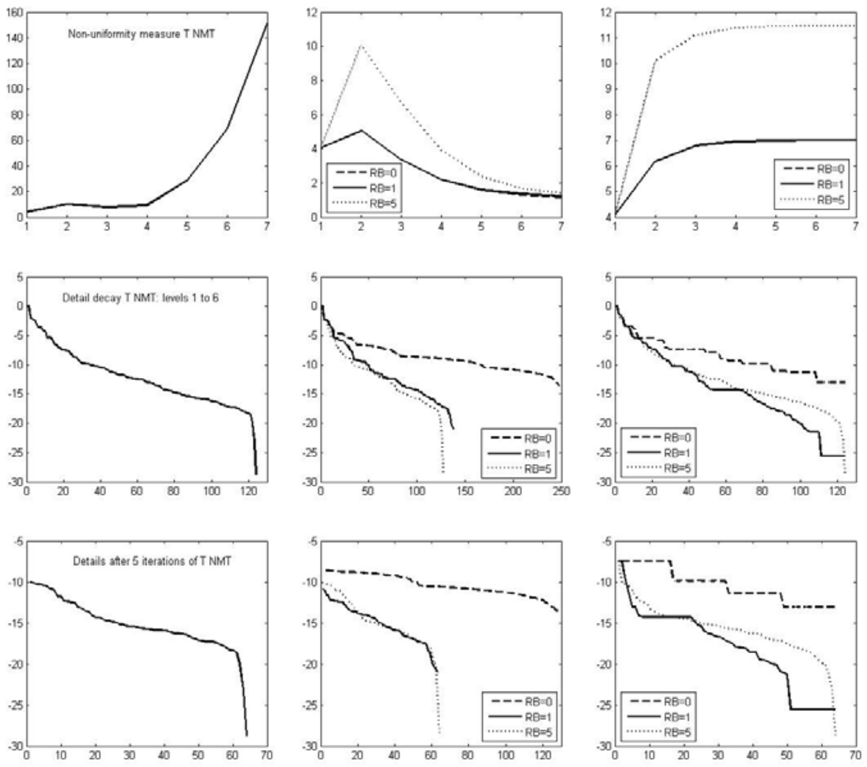


Fig. 2. Comparison among T normal MT (left), S_2/T normal MT (middle) and S_1/T normal MT (right), when performed on the circle setup from Fig 1. The upper row plots $\mathcal{N}(\mathbf{v}^{j-1})$ for $j = 1, 2, \ldots, 7$. The middle row plots the corresponding $\log_2$-values of $|d^j|$ for levels $j \leq 5$ in descending order. The lower row plots $\log_2(|d^5|)$.

increase the irregularity of the initial data. While the S_2/T normal MT is able to "recover" and after five more iterations decreases the non-uniformity measure to the level of the pure S_2 normal MT, this is not the case with the S_1/T normal MT. Indeed, applied to the circle example, S_1 prediction never improves $\mathcal{N}(\mathbf{v})$, since at vertices of the initial set $\mathbf{v}^0$ the local non-uniformity measure does not decrease for any $j \geq 1$. When $RB = 1$, the adaptive S_2/T, resp S_1/T normal MT applies S_2, resp. S_1, in all regions of highly non-uniform spacing. This explains the almost identical plots for S_2/T and S_2, resp. S_1/T and S_1, normal MTs for small j. As discussed before, $RB = 1$ corresponds to $\mathcal{N}(\mathbf{v}) \approx 2$ and, thus, the corresponding plots for S_2 and S_2/T normal MTs start to slightly differ only when the data is already close to regular. Moreover, once $\mathcal{N}(\mathbf{v}^j) < 2$ is achieved, the S_2/T normal MT with $RB = 1$ turns into pure T normal MT. This is confirmed by the $\log_2|d^5|$ plots which contain half as many data points for S_2/T as compared with S_2 normal MT. On the other hand, because of the inability of S_1 to improve the non-uniformity measure, for every level $j \in \mathbb{N}$

in the S_1/T normal MT there will be points (in our particular case it is one point when $RB = 5$ and two points when $RB = 1$), predicted via mid-point interpolation, which leads to the same ℓ_∞ norm for the details as in the case of pure S_1 normal MT. Hence the worst-case detail decay rate of the S_1/T adaptive scheme will be only 2, instead of 3.

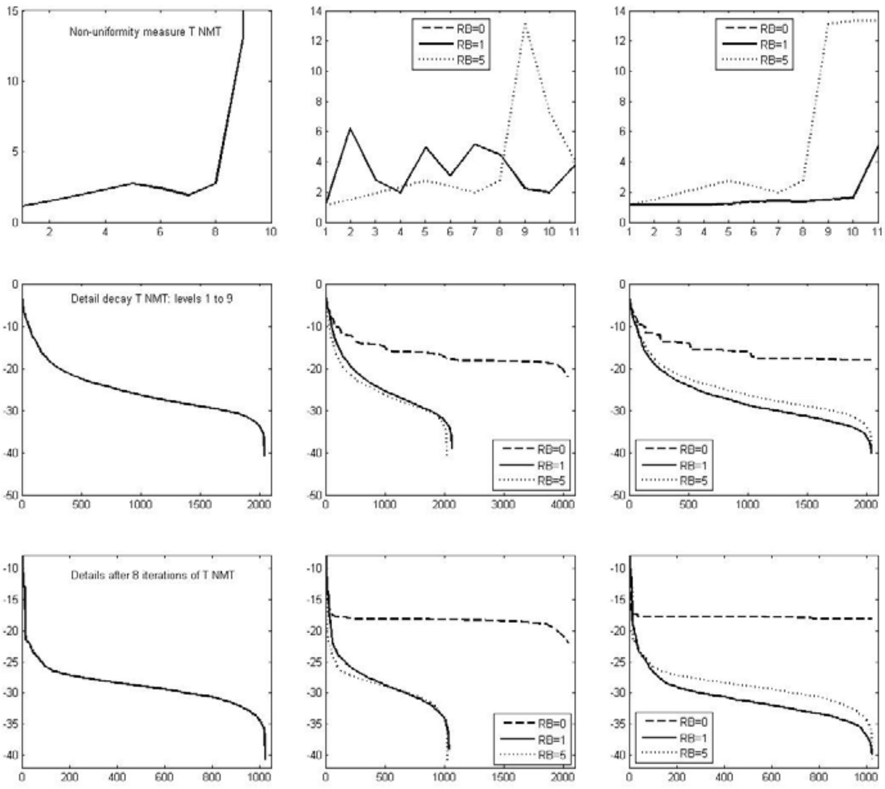


Fig. 3. Numerical results for twisted circle. Plots position is analogous to Fig. 2.

In the second example (twisted circle), we performed the same experiments. However, since the T normal MT brakes down only after 9 iterations, we take into account more steps and compare the details after 8 iterations, instead of 5. The results are summarized in Fig. 3. Since the twisted circle is composed of circle arcs, some of the phenomena observed on the first example appear here, too, but due to the low smoothness of $\mathcal{C}$, there are also additional effects. First of all, as we start with almost equally spaced points $\mathbf{v}^0$, the non-uniformity measure for T and S_1 normal MT stays low for small j, until the singularities of the initial curve are "detected". This happens only at level $j = 8$ because the initial points were chosen very close to the singularities. Then the non-uniformity

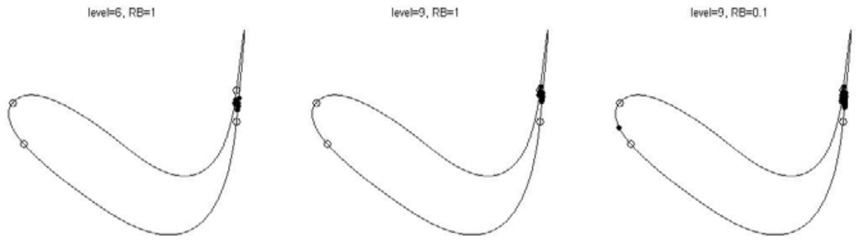


Fig. 4. The adaptive schemes detect singularities

measure $\mathcal{N}(\mathbf{v}^j)$ corresponding to the T normal MT jumps from 3 to 13, and the transform fails at the next level. Therefore, as can be seen from the graphs, for levels $j \leq 9$ both S_2/T and S_1/T normal MT with $RB = 5$ coincide with the pure T normal MT. After that S_2/T manages to improve the non-uniformity measure, while S_1/T just keeps it as it is.

On the other hand, as shown on Fig. 4, the S_2 normal MT detects the presence of singularities from the very beginning, although it cannot localize them as well as an interpolating normal MT would do. On a smooth curve with a single singularity point, we start with a $\mathbf{v}^0$ consisting of 4 irregularly spaced points, and show the locations where S_2/T normal MT still uses S_2, i.e., where the local non-uniformity measure exceeds RB. On the left plot, we see that at level $j = 5$ the irregularity of the initial data is essentially removed, and the only places, where S_2 prediction is used, is near the singularity. In the middle plot, we show how the localization of the singularity by the S_2/T normal MT improves after three more iterations. On the right plot, we see that even for very small values of RB (in the particular case $RB = 0.1$) this still works, even though only after slightly more iterations. The isolated S_2 interval away from the singularity disappears for $j = 10$. This cannot be achieved with the S_1/T normal MT, since it can never fully recover from the irregularity of the initial data and will use S_1 predictions near vertices in $\mathbf{v}^0$ for any j.

We conclude our experiments with a comparison between our S_2/T normal MT and the T_ω normal MT proposed in [3]. To the best of our knowledge, the latter is the only other normal MT that is well-defined, converges and (in case of smooth initial curve) after finitely many iterations turns into a pure T normal MT. The family T_ω is defined via

$$T_\omega = (1-\omega)S_1 + \omega T, \qquad 0 < \omega \leq 1,$$

and T_ω is proven to be weakly contractive (see [3] for details) with bound

$$R(\omega) = \frac{4}{\omega}\left(1 + \sqrt{1 - \frac{\omega}{2}} - 1\right) \tag{16}$$

for each specific $0 \leq \omega \leq 1$. Combined with Theorem 5.7 from [3], this implies the existence of an increasing sequence $0 < \omega_1 < \omega_2 < \ldots < \omega_J < 1 = \omega_{J+1} = \ldots$

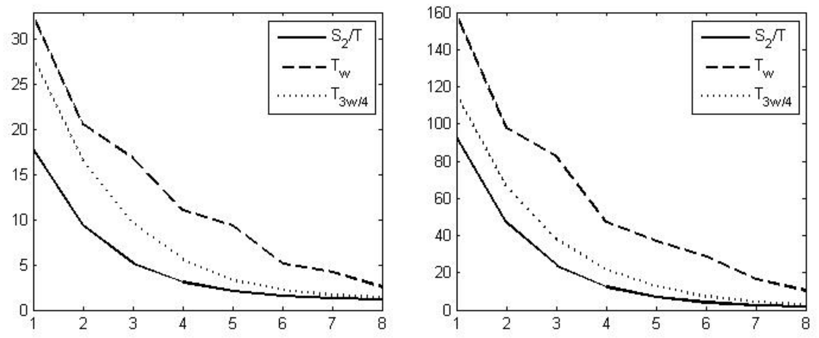


Fig. 5. Comparison between our adaptive scheme and the family T_ω proposed in [3]

such that (1) with T replaced by T_{ω_j} leads to a well-defined transform called T_ω normal MT (the ω_j as well as the level J after which the T_ω normal MT coincides with the T normal MT, depend on $\mathcal{C}$ and $\mathbf{v}^0$). All one has to do is to choose ω_j such that $\mathcal{N}(\mathbf{v}^{j-1}) \leq R(\omega_j)$ (concrete rules for picking the ω_j resp. a locally adaptive version of the T_ω normal MT have not been elaborated on in [3]).

Fig. 5 displays the decay of the non-uniformity measure $\mathcal{N}(\mathbf{v}^j)$ for $j \leq 8$ associated with S_2, T_ω, and $T_{3\omega/4}$ normal MT. Since the decay of the non-uniformity measure for smooth $\mathcal{C}$ is basically a property of the underlying subdivision operator, for this test we let $\mathcal{C}$ be a straight line, and $\mathbf{v}^0$ consist of randomly chosen points on it with large $\mathcal{N}(\mathbf{v}^0)$ value. For the T_ω normal MT, we always work with the largest possible value $\omega_j = 8\mathcal{N}(\mathbf{v}^{j-1})/(\mathcal{N}(\mathbf{v}^{j-1})+1)^2$ satisfying (16), while for the $T_{3\omega/4}$ normal MT we use $T_{3\omega_j/4}$ as prediction operator. In Fig. 5 we have plotted two examples. We observe that S_2 improves the non-uniformity measure faster (actually, for S_2 it can be straightforwardly verified that on a straight line $\mathcal{N}(\mathbf{v}^j) < (\mathcal{N}(\mathbf{v}^{j-1})+1)/2$, $j \in \mathbb{N}$), and allows us to switch to T at an earlier level. In the first example we can do this after three iterations, while we need four iterations if using $T_{3\omega/4}$ or seven iterations if using T_ω, and in the second example we can do this after five iterations, while we need six iterations if using $T_{3\omega/4}$ or more than eight if using T_ω). The other observation is that for both examples $T_{3\omega/4}$ normal MT performs better than T_ω normal MT, which indicates that, in order to minimize the fine-scale non-uniformity measure, the ω_j should be chosen carefully.

In this paper we followed the classical approach to normal MTs and worked with linear univariate subdivision schemes T as prediction operators. Another possible way to obtain globally well-posed and convergent normal MTs with higher detail decay rates for smooth curves is to consider geometry-based subdivision schemes T, e.g., those introduced in [4,11,15], as prediction operators.

References

1. Baraniuk, R., Janssen, M., Lavu, S.: Multiscale approximation of piecewise smooth two-dimensional functions using normal triangulated meshes. Appl. Comput. Harm. Anal. 19, 92–130 (2005)
2. Binev, P., Dahmen, W., DeVore, R.A., Dyn, N.: Adaptive approximation of curves. In: Approximation Theory, pp. 43–57. Acad. Publ. House, Sofia (2004)
3. Daubechies, I., Runborg, O., Sweldens, W.: Normal multiresolution approximation of curves. Constr. Approx. 20, 399–462 (2005)
4. Dyn, N., Floater, M., Hormann, K.: Four-point curve subdivision based on iterated chordal and centripetal parameterizations. Comp. Aided Geom. Design 26, 279–286 (2009)
5. Friedel, I., Khodakovsky, A., Schröder, P.: Variational normal meshes. ACM Trans. Graph. 23, 1061–1073 (2004)
6. Guskov, I., Vidimce, K., Sweldens, W., Schröder, P.: Normal meshes. In: Computer Graphics Proceedings (Siggraph 2000), pp. 95–102. ACM Press, New York (2000)
7. Harizanov, S., Oswald, P.: Stability of nonlinear subdivision and multiscale transforms. Constr. Approx. 31, 359–393 (2010)
8. Harizanov, S., Oswald, P., Shingel, T.: Normal multi-scale transforms for curves. Found. Comput. Math. (2009) (submitted)
9. Khodakovsky, A., Guskov, I.: Compression of normal meshes. In: Geometric Modeling for Scientific Visualization, pp. 189–207. Springer, Berlin (2003)
10. Lavu, S., Choi, H., Baraniuk, R.: Geometry compression of normal meshes using rate-distortion algorithms. In: Eurographics/ACM Siggraph Symposium on Geometry Processing, pp. 52–61. RWTH Aachen (2003)
11. Marinov, M., Dyn, N., Levin, D.: Geometrically controlled 4-point interpolatory schemes. In: Dodgson, N., Floater, M., Sabin, M. (eds.) Advances in Multiresolution for Geometric Modelling, pp. 301–315. Springer, Heidelberg (2005)
12. Oswald, P.: Normal multi-scale transforms for surfaces. This Proceedings (submitted)
13. Runborg, O.: Introduction to normal multiresolution analysis. In: Samelson, K. (ed.) Multiscale Methods in Science and Engineering. LNCS, vol. 44, pp. 205–224. Springer, Heidelberg (2005)
14. Runborg, O.: Fast interface tracking via a multiresolution representation of curves and surfaces. Commun. Math. Sci. 7, 365–389 (2009)
15. Sabin, M., Dodgson, N.: A circle-preserving variant of the four-point subdivision scheme. In: Dahlen, M., Morken, K., Schumaker, L.L. (eds.) Mathematical Methods for Curves and Surfaces: Tromso 2004, pp. 275–286. Nashboro Press (2005)

Helmholtz-Hodge Decomposition on $[0,1]^d$ by Divergence-Free and Curl-Free Wavelets

Souleymane Kadri Harouna and Valérie Perrier

Laboratoire Jean Kuntzmann, University of Grenoble,
and CNRS, BP 53 Grenoble Cedex 9, France
{Souleymane.Kadri-Harouna,Valerie.Perrier}@imag.fr

Abstract. This paper deals with the Helmholtz-Hodge decomposition of a vector field in bounded domain. We present a practical algorithm to compute this decomposition in the context of divergence-free and curl-free wavelets satisfying suitable boundary conditions. The method requires the inversion of divergence-free and curl-free wavelet Gram matrices. We propose an optimal preconditioning which allows to solve the systems with a small number of iterations. Finally, numerical examples prove the accuracy and the efficiency of the method.

Keywords: Divergence-free and curl-free wavelets, Helmholtz-Hodge decomposition.

1 Introduction

Vector field analysis is ubiquitous in engineering, physics or applied mathematics. Most of the solutions of problems arising from these domains are vector fields and they have some compatibility properties related to the nature of the problem. This is the case in the numerical simulation of incompressible fluid flows where the velocity field is divergence-free or in electromagnetism where the electric field contained in the electromagnetic field is curl-free.

The Helmholtz-Hodge decomposition, under certain smoothness assumptions, allows to separate any vector field into the sum of three uniquely defined components: divergence-free, curl-free and gradient of a harmonic function. Thus, the Helmholtz-Hodge decomposition provides a powerful tool for several applications such as the resolution of partial differential equations [14], aerodynamic design [25], detection of flow features [22] or computer graphics [21]. Therefore, it is important to have at hand an efficient algorithm to deal with such decomposition numerically.

In case of periodic boundary conditions, Fourier domain offers an ideal setting to compute the Helmholtz-Hodge decomposition, thanks to the Leray projector which writes explicitly [11]. For more general (physical) boundary conditions, this decomposition is usually achieved by solving a Poisson equation relative to each field component: this is the case when using finite element or finite difference methods [14]. The well known drawback of these methods is their cost, for example in particle-based physical simulations. Then, one resorts to mesh-less method [21] or methods leading to variational equations [22].

In the wavelet setting the Helmholtz-Hodge decomposition will not use the resolution of a Poisson equation, since one knows explicit bases for the divergence-free and curl-free function spaces [12,13,19,26]. In this context, Urban [27], and latter Deriaz and Perrier [11] introduced methods to compute the orthogonal projections onto divergence-free and curl-free wavelet bases. These wavelet methods provide accuracy for a small number of degrees of freedom, due to the good nonlinear approximation property provided by wavelet bases [7]. However, the works of [11,27] were limited to periodic boundary conditions for lack of suitable bases.

The main objective of this paper is to extend the works of [11,27] to more general physical boundary conditions. We first recall the principles of the tensor-product divergence-free and curl-free wavelet construction on the cube, that was detailed in [18] for the 2D case (see also [17]). Similar constructions, leading to different bases, were proposed by Stevenson in general dimension [23,24]. Then we propose an effective method for the Helmholtz-Hodge decomposition based on the computation and inversion of corresponding Gram matrices. The tensor structure of the bases is fully exploited to reduce the computational complexity. Moreover the system is solved with a low complexity, thanks to an optimal preconditioning.

The layout of this paper is as follows. In Section 2, we recall the theoretical definition and mathematical background of the Helmholtz-Hodge decomposition on a bounded domain of $\mathbb{R}^d$. In Section 3 we explicit the construction of divergence-free and curl-free wavelets on $[0,1]^d$ with desired boundary conditions. Section 4 is devoted to the description of the numerical method for the Helmholtz-Hodge decomposition, and numerical examples will illustrate its performance.

2 Helmholtz-Hodge Decomposition

We recall in this section some definitions related to the Helmholtz-Hodge decomposition on a bounded Lipschitz domain Ω of $\mathbb{R}^d$ [14]. We assume that the domain Ω and its boundary Γ have sufficient regularities (see [1,14]).

2.1 Definitions

The Helmholtz-Hodge decomposition theorem [6,14] states that any vector field $\mathbf{u} \in (L^2(\Omega))^d$ can be uniquely decomposed into the sum of its divergence-free, curl-free and gradient of harmonic function components:

$$\mathbf{u} = \mathbf{u}_{\text{div}} + \mathbf{u}_{\text{curl}} + \mathbf{u}_{\text{har}} \tag{1}$$

with

$$\nabla \cdot \mathbf{u}_{\text{div}} = 0 \qquad \text{and} \qquad \nabla \times \mathbf{u}_{\text{curl}} = 0. \tag{2}$$

The last component $\mathbf{u}_{\text{har}}$ is both divergence-free and irrotational:

$$\nabla \cdot \mathbf{u}_{\text{har}} = 0 \qquad \text{and} \qquad \nabla \times \mathbf{u}_{\text{har}} = 0. \tag{3}$$

Following [1,14], this decomposition may alternatively be written using scalar potentials. There exist a scalar potential $q \in H_0^1(\Omega)$ and a harmonic potential $h \in H^1(\Omega)$ such as

$$\mathbf{u}_{\text{curl}} = \nabla q \qquad \text{and} \qquad \mathbf{u}_{\text{har}} = \nabla h.$$

Moreover, the scalars q and h are uniquely defined.

In terms of spaces, the decomposition (1) corresponds to an orthogonal splitting of $(L^2(\Omega))^d$:

$$(L^2(\Omega))^d = \mathcal{H}_{div}(\Omega) \oplus \mathcal{H}_{curl}(\Omega) \oplus \mathcal{H}_{har}(\Omega) \tag{4}$$

where $\mathcal{H}_{div}(\Omega)$ is the space of divergence-free vector functions of $(L^2(\Omega))^d$ with vanishing normal boundary condition:

$$\mathcal{H}_{div}(\Omega) = \{\mathbf{u} \in (L^2(\Omega))^d : \ \mathbf{div}(\mathbf{u}) = 0, \ \mathbf{u} \cdot \mathbf{n}|_\Gamma = 0\}. \tag{5}$$

For $d = 2, 3$, the space $\mathcal{H}_{div}(\Omega)$ coincides with the *curl of potential* space (see [1,2,14]):

$$\mathcal{H}_{div}(\Omega) = \{\mathbf{u} = \mathbf{curl}(\boldsymbol{\Psi}) : \boldsymbol{\Psi} \in (H_0^1(\Omega))^d\}, \qquad (\boldsymbol{\Psi} \in H_0^1(\Omega) \ \text{ if } \ d = 2) \tag{6}$$

In this case, the component $\mathbf{u}_{\text{div}}$ of (1) reads $\mathbf{u}_{\text{div}} = \mathbf{curl}(\boldsymbol{\Psi})$.

On the other hand, the space $\mathcal{H}_{curl}$ corresponds to the gradient of $H^1(\Omega)$-potentials which vanish on Γ:

$$\mathcal{H}_{curl}(\Omega) = \{\mathbf{u} = \nabla q : \ q \in H_0^1(\Omega)\}. \tag{7}$$

Finally, $\mathcal{H}_{har}$ corresponds to the gradient of $H^1(\Omega)$-harmonic potentials:

$$\mathcal{H}_{har}(\Omega) = \{\mathbf{u} = \nabla h : \ h \in H^1(\Omega), \ \Delta h = 0\}. \tag{8}$$

Other splittings exist, for example in $(H_0^1(\Omega))^d$ to incorporate homogeneous boundary conditions [14].

In the whole space $\mathbb{R}^d$ or with periodic boundary conditions, the decomposition (1) is explicit in Fourier domain and the third term vanishes: $\mathbf{u}_{\text{har}} = 0$ (one obtains the Helmholtz decomposition in this case [11,27]). In the wavelet context, an iterative procedure was proposed by Deriaz and Perrier [11]. The purpose here is to extend such method with boundary conditions for the spaces $\mathcal{H}_{div}(\Omega)$, $\mathcal{H}_{curl}(\Omega)$ and $\mathcal{H}_{har}(\Omega)$ introduced in (5, 7, 8).

3 Divergence-Free and Curl-Free Wavelets on $[0,1]^d$

This section introduces the principles of the construction and main properties of divergence-free and curl-free wavelets on the hypercube $[0,1]^d$. The two dimensional case $d = 2$ has been detailed in [18], with explicit examples (spline, Daubechies wavelets), and practical tools for the implementation (filters, fast wavelet transform, ..). We develop below the extension to more general dimensions $d \geq 3$. Recently, Stevenson proposed a first construction [23,24], which led to alternative wavelet bases. A first advantage of our construction is that it fits perfectly with the classical multiresolution analysis construction on the interval $[0, 1]$.

The construction is based on 1D multiresolution analysis generators $(\varphi^1, \tilde{\varphi}^1)$ and $(\varphi^0, \tilde{\varphi}^0)$ linked by differentiation / integration, and introduced by Lemarié-Rieusset and collaborators in original works [16,19]. It follows two steps, described in the two forthcoming sections:

(i) Construction of two biorthogonal MRAs of $L^2(0, 1)$ linked by differentiation / integration.

(ii) Construction of MRAs and wavelet bases of $\mathcal{H}_{div}(\Omega)$ and $\mathcal{H}_{curl}(\Omega)$.

3.1 Multiresolution Analyses of $L^2(0, 1)$ Linked by Differentiation / Integration

The construction of regular biorthogonal multiresolution analyses (BMRA) on the interval $[0, 1]$ is now classical (see [5,9,15,20]). It begins with a pair of biorthogonal compactly supported scaling functions $(\varphi^1, \tilde{\varphi}^1)$ [8] of $L^2(\mathbb{R})$, with some r polynomial reproduction:

$$x^\ell = \sum_{k\in\mathbb{Z}} \langle x^\ell, \tilde{\varphi}^1(x-k)\rangle \; \varphi^1(x-k) \qquad \text{for} \qquad 0 \le \ell \le r-1, \tag{9}$$

and similarly for $\tilde{\varphi}^1$, with $\tilde{r}$ polynomial reproduction.

Following classical constructions, one defines finite dimensional biorthogonal multiresolution spaces

$$V_j^1 = \text{span}\{\varphi_{j,k}^1 \;;\; 0 \le k \le N_j - 1\} \;\text{ and }\; \tilde{V}_j^1 = \text{span}\{\tilde{\varphi}_{j,k}^1 \;;\; 0 \le k \le N_j - 1\} \tag{10}$$

whose dimension $N_j \simeq 2^j$ depends on some *free* integer parameters (δ_0, δ_1). The scaling functions $\varphi_{j,k}^1$ satisfy $\varphi_{j,k}^1 = 2^{j/2}\varphi^1(2^j x - k)$ "inside" the interval $[0, 1]$, but this is no more true near the boundaries 0 and 1 (idem for $\tilde{\varphi}_{j,k}^1$). In practice, the scale index j must be great than some index j_{min}, to avoid boundary effects. The biorthogonality between bases is $< \varphi_{j,k}^1 / \tilde{\varphi}_{j,k'}^1 >= \delta_{k,k'}$.

The approximation order provided by such MRA (V_j^1) in $L^2(0, 1)$ is r:

$$\forall\; f \in H^s(0, 1), \qquad \inf_{f_j \in V_j^1} \|f - f_j\|_{L^2(0,1)} \le C 2^{-js}, \;\; 0 \le s \le r \tag{11}$$

whereas $(\tilde{V}_j^1)$ has approximation order $\tilde{r}$.

Homogeneous Dirichlet boundary conditions can be simply imposed on (V_j^1) by removing one scaling function at each boundary 0 and 1:

$$V_j^D = V_j^1 \cap H_0^1(0, 1) = \text{span}\{\varphi_{j,k}^1 \;;\; 1 \le k \le N_j - 2\}. \tag{12}$$

A possibility to adjust the dimension of the two biorthogonal spaces $(V_j^D, \tilde{V}_j^D)$ is to impose $\tilde{V}_j^D = \tilde{V}_j^1 \cap H_0^1(0, 1)$, and $\tilde{V}_j^D$ becomes (keeping the same notation for the basis functions, which have changed after biorthogonalization):

$$\tilde{V}_j^D = \text{span}\{\tilde{\varphi}^1_{j,k}\ ;\ 1 \le k \le N_j - 2\}. \tag{13}$$

As usual, biorthogonal wavelet spaces $(W_j^1, \tilde{W}_j^1)$ are defined by

$$W_j^1 = V_{j+1}^1 \cap (\tilde{V}_j^1)^\perp \qquad \tilde{W}_j^1 = \tilde{V}_{j+1}^1 \cap (V_j^1)^\perp \tag{14}$$

and generated by finite dimensional wavelet bases on the interval [15,20]:

$$W_j^1 = \text{span}\{\psi^1_{j,k}\ ;\ 0 \le k \le 2^j - 1\} \text{ and } \tilde{W}_j^1 = \text{span}\{\tilde{\psi}^1_{j,k}\ ;\ 0 \le k \le 2^j - 1\}. \tag{15}$$

The difficulty now is to derive a new biorthogonal MRA $(V_j^0, \tilde{V}_j^0)$ of $L^2(0,1)$ such that

$$\frac{d}{dx} V_j^1 = V_j^0$$

The existence of such biorthogonal MRA was already proved by Jouini and Lemarié-Rieusset [16] and it should be based on generators $(\varphi^0, \tilde{\varphi}^0)$ introduced in [19] satisfying

$$(\varphi^1(x))' = \varphi^0(x) - \varphi^0(x-1) \qquad \text{and} \qquad (\tilde{\varphi}^0(x))' = \tilde{\varphi}^1(x+1) - \tilde{\varphi}^1(x). \tag{16}$$

Let us introduce the primitive space for $\tilde{V}_j^1$:

$$\int_0^x \tilde{V}_j^1 = \{g\ : \exists\ f \in \tilde{V}_j^1 \text{ such that } g(x) = \int_0^x f(t)dt\}.$$

In [18], we proposed a practical construction of spaces $(V_j^0, \tilde{V}_j^0)$:

$$V_j^0 = \text{span}\{\varphi^0_{j,k}\ ;\ 0 \le k \le N_j - 2\} \text{ and } \tilde{V}_j^0 = \text{span}\{\tilde{\varphi}^0_{j,k}\ ;\ 0 \le k \le N_j - 2\} \tag{17}$$

different from the underlying spaces of [24], but satisfying the following proposition.

Proposition 1. *The two BMRAs $(V_j^\epsilon, \tilde{V}_j^\epsilon)_{\epsilon=0,1}$ of $L^2(0,1)$ constructed in [18] from biorthogonal generators $(\varphi^\epsilon, \tilde{\varphi}^\epsilon)_{\epsilon=0,1}$ satisfying relation (16), verify*

$$(i)\ \frac{d}{dx} V_j^1 = V_j^0 \quad \textit{and} \quad \frac{d}{dx} \circ \mathcal{P}_j^1 f = \mathcal{P}_j^0 \circ \frac{d}{dx} f, \quad \forall\ f \in H^1(0,1)$$

$$(ii)\ \tilde{V}_j^0 = H_0^1(0,1) \cap \int_0^x \tilde{V}_j^1 \quad \textit{and} \quad \frac{d}{dx} \circ \tilde{\mathcal{P}}_j^0 f = \tilde{\mathcal{P}}_j^1 \circ \frac{d}{dx} f, \quad \forall\ f \in H_0^1(0,1),$$

where $(\mathcal{P}_j^\epsilon, \tilde{\mathcal{P}}_j^\epsilon)$ are the biorthogonal projectors on $(V_j^\epsilon, \tilde{V}_j^\epsilon)$.

Wavelet bases of the biorthogonal MRA $(V_j^0, \tilde{V}_j^0)_{j \ge j_{min}}$ are simply defined by respectively differentiating and integrating the wavelets of $(V_j^1, \tilde{V}_j^1)_{j \ge j_{min}}$, as stated by the following proposition [16,18].

Proposition 2. *[16,18] Let $\{\psi^1_{j,k}\}$ and $\{\tilde{\psi}^1_{j,k}\}$ be biorthogonal wavelet bases of respectively W^1_j and $\tilde{W}^1_j$. Then, the wavelets defined by*

$$\psi^0_{j,k} = 2^{-j}(\psi^1_{j,k})' \qquad \text{and} \qquad \tilde{\psi}^0_{j,k} = -2^j \int_0^x \tilde{\psi}^1_{j,k} \tag{18}$$

are respectively biorthogonal wavelet bases of W^0_j and $\tilde{W}^0_j$:

$$W^0_j = V^0_{j+1} \cap (\tilde{V}^0_j)^\perp \qquad \text{and} \qquad \tilde{W}^0_j = \tilde{V}^0_{j+1} \cap (V^0_j)^\perp. \tag{19}$$

Remark 1. The wavelets $\psi^0_{j,k}$ and $\tilde{\psi}^0_{j,k}$ defined by (18) differ from the standard wavelet construction [5,9,20] in BMRA $(V^0_j, \tilde{V}^0_j)$: indeed, the usual wavelets on the interval do not lead to the differentiation/integration relation (18), except for interior wavelets.

3.2 Divergence-Free Scaling Functions and Wavelets on $[0,1]^d$

Le Ω be the hypercube $\Omega = [0,1]^d$. The objective in this section is to derive wavelet bases of the space $\mathcal{H}_{div}(\Omega)$, with vanishing outward normal at the boundary $\Gamma = \partial\Omega$. Following (5), $\mathcal{H}_{div}(\Omega)$ is the curl of the space $H^1_0(\Omega)$ $(d=2)$ or $(H^1_0(\Omega))^d$ $(d=3)$ of scalar (vector for $d=3$) stream functions [1]. We begin with the description of basis functions in the 2D case (already detailed in [18]), then we propose a generalization of the construction for $d \geq 3$.

Two-Dimensional Case: We start with the 2D MRA $(V^D_j \otimes V^D_j)_{j \geq j_{min}}$ of $H^1_0(\Omega)$, where $(V^D_j)_{j \geq j_{min}}$ is the 1D MRA of $H^1_0(0,1)$ defined in Section 3.1 (12). For each scale index $j \geq j_{min}$, divergence-free scaling functions on $\Omega = [0,1]^2$ are constructed by taking the curl of scaling functions of $V^D_j \otimes V^D_j$:

$$\Phi^{div}_{j,\mathbf{k}} := \mathbf{curl}[\varphi^D_{j,k_1} \otimes \varphi^D_{j,k_2}] = \begin{vmatrix} \varphi^D_{j,k_1} \otimes (\varphi^D_{j,k_2})' \\ \\ -(\varphi^D_{j,k_1})' \otimes \varphi^D_{j,k_2} \end{vmatrix}, \quad 1 \leq k_1, k_2 \leq N_j - 2. \tag{20}$$

The choice of space V^D_j ensures that the divergence-free scaling functions satisfy the boundary condition $\Phi^{div}_{j,\mathbf{k}} \cdot \mathbf{n} = 0$ by construction. Let $\mathbf{V}^{div}_j$ be the space spanned by these divergence-free scaling functions:

$$\mathbf{V}^{div}_j = \text{span}\{\Phi^{div}_{j,\mathbf{k}}\}, \quad 1 \leq k_1, k_2 \leq N_j - 2. \tag{21}$$

By construction, the spaces $\mathbf{V}^{div}_j$ form a multiresolution analysis of $\mathcal{H}_{div}(\Omega)$, since it can be proven from Proposition 1 that we have [18]:

$$\mathbf{V}^{div}_j = (V^D_j \otimes V^0_j) \times (V^0_j \otimes V^D_j) \cap \mathcal{H}_{div}(\Omega). \tag{22}$$

In the same manner, the corresponding anisotropic divergence-free wavelets on Ω are defined by taking the curl of the three types of scalar anisotropic wavelets associated to $V^D_j \otimes V^D_j$: for $\mathbf{j} = (j_1, j_2)$, with $j_1, j_2 > j_{min}$,

$$\Psi^{div,1}_{\mathbf{j},\mathbf{k}} := \mathbf{curl}[\varphi^D_{j_{min},k} \otimes \psi^D_{j_2,k_2}],\ 1 \le k \le N_{j_{min}} - 2,\quad 0 \le k_2 \le 2^{j_2} - 1$$
$$\Psi^{div,2}_{\mathbf{j},\mathbf{k}} := \mathbf{curl}[\psi^D_{j_1,k_1} \otimes \varphi^D_{j_{min},k}],\ 0 \le k_1 \le 2^{j_1} - 1,\quad 1 \le k \le N_{j_{min}} - 2$$
$$\Psi^{div,3}_{\mathbf{j},\mathbf{k}} := \mathbf{curl}[\psi^D_{j_1,k_1} \otimes \psi^D_{j_2,k_2}],\ 0 \le k_1 \le 2^{j_1} - 1,\ 0 \le k_2 \le 2^{j_2} - 1,$$

$(\psi^D_{j,k})$ being the wavelet basis of $W^D_j = V^D_{j+1} \cap (\tilde{V}^D_j)^\perp$.

Three-Dimensional Case: Divergence-free scaling functions and wavelets on $\Omega = [0,1]^3$ are constructed by taking the curl of suitable scaling functions and wavelets of $(H^1(\Omega))^3$ [1,14]. We will focus on the scaling function construction, since the same technique is used for wavelets. The divergence-free scaling functions are defined by

$$\Phi^{div}_{1,j,\mathbf{k}} := \mathbf{curl}\left|\begin{matrix} 0 \\ 0 \\ \varphi^D_{j,k_1} \otimes \varphi^D_{j,k_2} \otimes \varphi^0_{j,k_3} \end{matrix}\right. = \left|\begin{matrix} \varphi^D_{j,k_1} \otimes (\varphi^D_{j,k_2})' \otimes \varphi^0_{j,k_3} \\ -(\varphi^D_{j,k_1})' \otimes \varphi^D_{j,k_2} \otimes \varphi^0_{j,k_3} \\ 0 \end{matrix}\right. \tag{23}$$

$$\Phi^{div}_{2,j,\mathbf{k}} := \mathbf{curl}\left|\begin{matrix} \varphi^0_{j,k_1} \otimes \varphi^D_{j,k_2} \otimes \varphi^D_{j,k_3} \\ 0 \\ 0 \end{matrix}\right. = \left|\begin{matrix} 0 \\ \varphi^0_{j,k_1} \otimes \varphi^D_{j,k_2} \otimes (\varphi^D_{j,k_3})' \\ -\varphi^0_{j,k_1} \otimes (\varphi^D_{j,k_2})' \otimes \varphi^D_{j,k_3} \end{matrix}\right. \tag{24}$$

$$\Phi^{div}_{3,j,\mathbf{k}} := \mathbf{curl}\left|\begin{matrix} 0 \\ \varphi^D_{j,k_1} \otimes \varphi^0_{j,k_2} \otimes \varphi^D_{j,k_3} \\ 0 \end{matrix}\right. = \left|\begin{matrix} -\varphi^D_{j,k_1} \otimes \varphi^0_{j,k_2} \otimes (\varphi^D_{j,k_3})' \\ 0 \\ (\varphi^D_{j,k_1})' \otimes \varphi^0_{j,k_2} \otimes \varphi^D_{j,k_3} \end{matrix}\right. \tag{25}$$

These functions are contained in $\mathcal{H}_{div}(\Omega)$ by construction. Let $\mathbf{V}^{div}_j$ be the space spanned by this family: $\mathbf{V}^{div}_j = span\{\Phi^{div}_{1,j,\mathbf{k}}, \Phi^{div}_{2,j,\mathbf{k}}, \Phi^{div}_{3,j,\mathbf{k}}\}$.

Since $\mathcal{H}_{div}(\Omega) = curl(H^1_0(\Omega))^3$, $\mathbf{V}^{div}_j$ is no more than the intersection of the following standard BMRA of $(L^2(\Omega))^3$:

$$\mathbf{V}_j = \left(V^1_j \otimes V^0_j \otimes V^0_j\right) \times \left(V^0_j \otimes V^1_j \otimes V^0_j\right) \times \left(V^0_j \otimes V^0_j \otimes V^1_j\right) \tag{26}$$

with $\mathcal{H}_{div}(\Omega)$.

To each scaling function, we can associate 7 types of anisotropic divergence-free generating wavelets by taking respectively the curl of wavelets of $\{0\} \times \{0\} \times (V^D_j \otimes V^D_j \otimes V^0_j)$, $(V^0_j \otimes V^D_j \otimes V^D_j) \times \{0\} \times \{0\}$ and $\{0\} \times (V^D_j \otimes V^0_j \otimes V^D_j) \times \{0\}$, among which we can extract a wavelet basis.

The construction may extend to larger dimensions $d \ge 3$ in the same way. As in the isotropic construction of Lemarié-Rieusset [19], we obtain in this case d types of divergence-free scaling functions For $1 \le i \le d$, the general formula of these scaling functions is given by

$$\Phi^{div}_{i,j,\mathbf{k}} := \begin{array}{r} \\ \\ \\ \text{row } i \;\rightarrow \\ \text{row } i+1 \rightarrow \\ \\ \\ \\ \end{array} \left| \begin{array}{l} 0 \\ \vdots \\ 0 \\ \varphi^0_{j,k_1} \otimes \cdots \otimes \varphi^0_{j,k_{i-1}} \otimes \varphi^D_{j,k_i} \otimes (\varphi^D_{j,k_{i+1}})' \otimes \cdots \otimes \varphi^0_{j,k_d} \\ -\varphi^0_{j,k_1} \otimes \cdots \otimes (\varphi^D_{j,k_i})' \otimes \varphi^D_{j,k_{i+1}} \otimes \varphi^0_{j,k_{i+2}} \otimes \cdots \otimes \varphi^0_{j,k_d} \\ 0 \\ \vdots \\ 0 \end{array} \right. \tag{27}$$

(for $i = d$, replace row $i+1$ by row d). These scaling functions $\Phi^{div}_{i,j,\mathbf{k}}$ satisfy the boundary condition : $\Phi^{div}_{i,j,\mathbf{k}} \cdot \mathbf{n} = 0$, by construction. The space $\mathbf{V}^{div}_j$ spanned by this family is included into the following vector-valued multiresolution analysis of $(L^2(\Omega))^d$:

$$\mathbf{V}_j = V^{(1)}_j \times \cdots \times V^{(d)}_j \quad \text{with} \quad V^{(i)}_j = V^{\delta_{1,i}}_j \otimes \cdots \otimes V^{\delta_{d,i}}_j, \;\; 1 \le i \le d \tag{28}$$

where $\delta_{j,i}$ denotes the Kronecker symbol. The corresponding wavelet family, generated by the $d(2^d-1)$ types of corresponding divergence-free wavelets, constitutes an alternative family to the basis built in Stevenson's work [24].

3.3 Curl-Free Scaling Functions and Wavelets on $[0,1]^d$

The construction of irrotational scaling functions and wavelets is easier than in the case of divergence-free functions, since it does not depend on the dimension d. According to the definition (7) of $\mathcal{H}_{curl}(\Omega)$, basis functions will be constructed by taking the gradient of scaling functions and wavelets of a multiresolution analysis of $H^1_0(\Omega)$.

The starting point is again a regular multiresolution analysis of $H^1_0(\Omega)$ given by d tensor-product of V^D_j:

$$\mathbf{V}_j = V^D_j \otimes \cdots \otimes V^D_j. \tag{29}$$

Then, the curl-free scaling functions of $\mathcal{H}_{curl}(\Omega)$ are defined by

$$\Phi^{\nabla}_{j,\mathbf{k}} = \nabla[\varphi^D_{j,k_1} \otimes \cdots \otimes \varphi^D_{j,k_d}] \qquad \text{and} \qquad \mathbf{V}^{\nabla}_j = \text{span}\{\Phi^{\nabla}_{j,\mathbf{k}}\}, \tag{30}$$

where $1 \le k_i \le N_j - 2$ for $1 \le i \le d$. By construction we have

$$\mathbf{V}^{\nabla}_j = \nabla[V^D_j \otimes \cdots \otimes V^D_j].$$

The multiresolution decomposition of the space $\mathbf{V}^{\nabla}_j$ leads to:

$$\mathbf{V}^{\nabla}_j = \nabla \left[V^D_{j_{min}} \otimes \cdots \otimes V^D_{j_{min}} \bigoplus_{j_{min} \le j_i \le j-1} \left(\sum_{\omega \in \Omega^*_d} W^{\omega}_{\mathbf{j}} \right) \right], \;\; 1 \le i \le d \tag{31}$$

with

$$\Omega_d^* = \{0,1\}^d \setminus (0,\cdots,0) \qquad \text{and} \qquad \mathbf{W}_{\mathbf{j}}^{\omega} = W_{j_1}^{\omega_1} \otimes \cdots \otimes W_{j_d}^{\omega_d},$$

where the spaces $W_{j_i}^{\omega_i}$ correspond to

$$W_{j_i}^{\omega_i} = W_{j_i}^{D} \ \text{ if } \ \omega_i = 1, \qquad W_{j_i}^{\omega_i} = V_{j_{min}}^{D} \ \text{ if } \ \omega_i = 0.$$

Denoting by $\Psi_{\mathbf{j},\mathbf{k}}^{\omega}$ the wavelets of $\mathbf{W}_{\mathbf{j}}^{\omega}$, we define the curl-free wavelets and spaces by

$$\Psi_{\mathbf{j},\mathbf{k}}^{\omega,\nabla} = \nabla\ [\Psi_{\mathbf{j},\mathbf{k}}^{\omega}] \qquad \text{and} \qquad \mathbf{W}_{\mathbf{j}}^{\omega,\nabla} = \text{span}\{\Psi_{\mathbf{j},\mathbf{k}}^{\omega,\nabla}\} \tag{32}$$

where $j_i \geq j_{min}$ and $0 \leq k_i \leq 2^j - 1$, for $1 \leq i \leq d$.

From Proposition 1, the spaces spanned by these curl-free functions are contained in the following BMRA of $(L^2(\Omega))^d$:

$$\mathbf{V}_j^{+} = \mathbf{V}_j^{1} \times \cdots \times \mathbf{V}_j^{d} \quad \text{with} \quad \mathbf{V}_j^{i} = V_j^{1-\delta_{1,i}} \otimes \cdots \otimes V_j^{1-\delta_{d,i}}, \ \ 1 \leq i \leq d$$

$\delta_{i,j}$ denotes the Kronecker symbol. This property allows fast coefficient computations on irrotational bases.

Since the spaces $\mathbf{V}_j$ defined in (29) constitute a multiresolution analysis of $H_0^1(\Omega)$, we get

$$H_0^1(\Omega) = \mathbf{V}_{j_{min}} \bigoplus_{j_i \geq j_{min}} \left(\sum_{\omega \in \Omega_d^*} \mathbf{W}_{\mathbf{j}}^{\omega} \right), \ \ 1 \leq i \leq d. \tag{33}$$

Taking the gradient of relation (33) and using again Proposition 1, we obtain

$$\mathcal{H}_{curl}(\Omega) = \mathbf{V}_{j_{min}}^{\nabla} \bigoplus_{j_i \geq j_{min}} \left(\sum_{\omega \in \Omega_d^*} \mathbf{W}_{\mathbf{j}}^{\omega,\nabla} \right), \ \ 1 \leq i \leq d. \tag{34}$$

This relation (34) was proved in [18] in the case of 2D construction.

4 Wavelet Helmholtz-Hodge Decomposition

4.1 Description of the Method

The Helmholtz-Hodge decomposition, introduced in Section 2, provides the orthogonal splitting of any vector field $\mathbf{u} \in (L^2(\Omega))^d$ into a divergence-free part, a curl-free part, and a gradient of a harmonic function:

$$\mathbf{u} = \mathbf{u}_{\text{div}} + \mathbf{u}_{\text{curl}} + \mathbf{u}_{\text{har}}, \tag{35}$$

where

$$\nabla \cdot \mathbf{u}_{\text{div}} = 0 \qquad \text{and} \qquad \mathbf{u}_{\text{div}} \cdot \mathbf{n} = 0$$

$$\nabla \times \mathbf{u}_{\text{curl}} = 0 \qquad \text{and} \qquad \mathbf{u}_{\text{curl}} \cdot \tau = 0$$

$$\nabla \times \mathbf{u}_{\text{har}} = 0 \qquad \text{and} \qquad \nabla \cdot \mathbf{u}_{\text{har}} = 0.$$

$\mathbf{n}$ and τ are respectively the unit outward normal and tangent to the boundary $\partial\Omega$.

Our aim in this section is to describe a practical way to compute the components $\mathbf{u}_{\text{div}}$ and $\mathbf{u}_{\text{curl}}$ when $\Omega = [0,1]^3$. The most natural way, already followed in [23] for the Helmholtz decomposition, is to use the divergence-free and curl-free scaling functions and wavelet bases constructed in the previous section. Since $\mathcal{H}_{div}(\Omega) = \text{span}\{\Psi^{div}_{\mathbf{j},\mathbf{k}}\}$ and $\mathcal{H}_{curl}(\Omega) = \text{span}\{\Psi^{\nabla}_{\mathbf{j},\mathbf{k}}\}$ (we adopt a unified notation for the wavelet bases), the components $\mathbf{u}_{\text{div}}$ and $\mathbf{u}_{\text{curl}}$ are searched under the form of their wavelet series:

$$\mathbf{u}_{\text{div}} = \sum_{\mathbf{j},\mathbf{k}} d^{\text{div}}_{\mathbf{j},\mathbf{k}}\ \Psi^{div}_{\mathbf{j},\mathbf{k}} \qquad \text{and} \qquad \mathbf{u}_{\text{curl}} = \sum_{\mathbf{j},\mathbf{k}} d^{\nabla}_{\mathbf{j},\mathbf{k}}\ \Psi^{\nabla}_{\mathbf{j},\mathbf{k}}. \tag{36}$$

By orthogonality of the decomposition (35) in $(L^2(\Omega))^d$, we obtain

$$\langle \mathbf{u}, \Psi^{div}_{\mathbf{j},\mathbf{k}}\rangle = \langle \mathbf{u}_{\text{div}}, \Psi^{div}_{\mathbf{j},\mathbf{k}}\rangle \qquad \text{and} \qquad \langle \mathbf{u}, \Psi^{\nabla}_{\mathbf{j},\mathbf{k}}\rangle = \langle \mathbf{u}_{\text{curl}}, \Psi^{\nabla}_{\mathbf{j},\mathbf{k}}\rangle. \tag{37}$$

Accordingly the computation of coefficients $(d^{\text{div}}_{\mathbf{j},\mathbf{k}})$ and $(d^{\nabla}_{\mathbf{j},\mathbf{k}})$ is reduced to the resolution of two linear systems

$$\mathbb{M}_{\text{div}}(d^{\text{div}}_{\mathbf{j},\mathbf{k}}) = (\langle \mathbf{u}, \Psi^{div}_{\mathbf{j},\mathbf{k}}\rangle) \qquad \text{and} \qquad \mathbb{M}_{\text{curl}}(d^{\nabla}_{\mathbf{j},\mathbf{k}}) = (\langle \mathbf{u}, \Psi^{\nabla}_{\mathbf{j},\mathbf{k}}\rangle), \tag{38}$$

where $\mathbb{M}_{\text{div}}$ and $\mathbb{M}_{\text{curl}}$ are respectively the Gram matrices of the bases $\{\Psi^{div}_{\mathbf{j},\mathbf{k}}\}$ and $\{\Psi^{\nabla}_{\mathbf{j},\mathbf{k}}\}$.

The above method is nothing but orthogonal projections from $(L^2(\Omega))^d$ to $\mathcal{H}_{div}(\Omega)$ and $\mathcal{H}_{curl}(\Omega)$ respectively. In practice, $\mathbf{u}_{\text{div}}$ is searched as $\mathbf{u}^j_{\text{div}} \in \mathbf{V}^{div}_j$ for some j, and $\mathbf{u}_{\text{curl}}$ as $\mathbf{u}^j_{\text{curl}} \in \mathbf{V}^{curl}_j$. Then we recover from the usual Jackson-type estimations:

$$\forall\ \mathbf{u} \in \mathcal{H}_{div}(\Omega) \cap (H^s(\Omega))^d, \quad \|\mathbf{u} - \mathbf{u}^j_{\text{div}}\|_{(L^2(\Omega))^d} \le C2^{-js}, \quad 0 \le s \le r-1,$$

and

$$\forall\ \mathbf{u} \in \mathcal{H}_{curl}(\Omega) \cap (H^s(\Omega))^d, \quad \|\mathbf{u} - \mathbf{u}^j_{\text{curl}}\|_{(L^2(\Omega))^d} \le C2^{-js}, \quad 0 \le s \le r-1,$$

where r denotes the approximation order provided by the generator φ^1.

The last component $\mathbf{u}_{\text{har}}$ of the decomposition (35) is computed by subtracting $\mathbf{u}_{\text{div}}$ and $\mathbf{u}_{\text{curl}}$ from $\mathbf{u}$:

$$\mathbf{u}_{\text{har}} = \mathbf{u} - \mathbf{u}_{\text{div}} - \mathbf{u}_{\text{curl}}. \tag{39}$$

4.2 Divergence-Free and Curl-Free Gram Matrices Computation

In this section we present a practical computation of matrices $\mathbb{M}_{\mathrm{div}}$ and $\mathbb{M}_{\mathrm{curl}}$. For easy reading, we focus on the matrix $\mathbb{M}_{\mathrm{curl}}$ in the 2D case. The extension to larger dimensions $d > 2$ follows readily from this two-dimensional case.

The key idea is to use the tensor structure of $\mathbb{M}_{\mathrm{curl}}$ to reduce the computation. Let $\mathbf{M}_j$ and $\mathbf{R}_j$ denote respectively the Gram and stiffness matrices of the 1D basis $\{\psi_{j,k}^D\}$:

$$[\mathbf{M}_j]_{k,k'} = \langle \psi_{j,k}^D, \psi_{j,k'}^D \rangle \qquad \text{and} \qquad [\mathbf{R}_j]_{k,k'} = \langle (\psi_{j,k}^D)', (\psi_{j,k'}^D)' \rangle \tag{40}$$

The tensor structure of the basis $\{\Psi_{\mathbf{j},\mathbf{k}}^{\nabla}\}$ allows to express the inner product $\langle \Psi_{\mathbf{j},\mathbf{k}}^{\nabla}, \Psi_{\mathbf{j}',\mathbf{k}'}^{\nabla} \rangle$ in terms of matrix elements (40). By definition of the basis functions we get

$$\langle \Psi_{\mathbf{j},\mathbf{k}}^{\nabla}, \Psi_{\mathbf{j}',\mathbf{k}'}^{\nabla} \rangle = \langle (\psi_{j_1,k_1}^D)' \otimes \psi_{j_2,k_2}^D, (\psi_{j_1',k_1'}^D)' \otimes \psi_{j_2',k_2'}^D \rangle + \langle \psi_{j_1,k_1}^D \otimes (\psi_{j_2,k_2}^D)', \psi_{j_1',k_1'}^D \otimes (\psi_{j_2',k_2'}^D)' \rangle$$

which becomes

$$\langle \Psi_{\mathbf{j},\mathbf{k}}^{\nabla}, \Psi_{\mathbf{j}',\mathbf{k}'}^{\nabla} \rangle = [\mathbf{M}_j]_{k_1,k_1'} \cdot [\mathbf{R}_j]_{k_2,k_2'} + [\mathbf{R}_j]_{k_1,k_1'} \cdot [\mathbf{M}_j]_{k_2,k_2'}. \tag{41}$$

Then $\mathbb{M}_{\mathrm{curl}}$ can be decomposed as

$$\mathbb{M}_{\mathrm{curl}} = \mathbf{M}_j \otimes \mathbf{R}_j + \mathbf{R}_j \otimes \mathbf{M}_j \tag{42}$$

The tensorial decomposition (42) has for main interest to reduce a 2D matrix-vector product with $\mathbb{M}_{\mathrm{curl}}$ to matrix-matrix products with $\mathbf{M}_j$ and $\mathbf{R}_j$. More precisely, if $(d_{\mathbf{j},\mathbf{k}}^{\nabla})$ denotes the vector of curl-free wavelet coefficients of $\mathbf{u}_{\mathrm{curl}}$, defined in (36), equation (42) leads to

$$[\mathbb{M}_{\mathrm{curl}}(d_{\mathbf{j},\mathbf{k}}^{\nabla})] = \mathbf{M}_j [d_{\mathbf{j},\mathbf{k}}^{\nabla}] \mathbf{R}_j + \mathbf{R}_j [d_{\mathbf{j},\mathbf{k}}^{\nabla}] \mathbf{M}_j, \tag{43}$$

where $[d_{\mathbf{j},\mathbf{k}}^{\nabla}]$ denotes the matrix of elements $d_{\mathbf{j},\mathbf{k}}^{\nabla}$. In practice the matrices only needed to compute and to store are the 1D matrices $\mathbf{M}_j$ and $\mathbf{R}_j$.

Finally, the matrix $\mathbb{M}_{\mathrm{curl}}$ has a sparse structure, due to the compact support of basis functions. Figure 1 shows the shape of $\mathbb{M}_{\mathrm{curl}}$, for $j = 6$, in the case of Daubechies generators with $r = 3$ vanishing moments. We remark that in 2D $\mathbb{M}_{\mathrm{div}} = \mathbb{M}_{\mathrm{curl}}$ (which is also the stiffness matrix of the Laplacian onto the wavelet basis $\{\psi_{j_1,k_1}^D \otimes \psi_{j_2,k_2}^D\}$), since we have

$$\forall\ \mathbf{u}, \mathbf{v} \in H_0^1(\Omega); \quad \int_{\Omega} \mathbf{curl}(\mathbf{u}) \cdot \mathbf{curl}(\mathbf{v})\, dx \ = \ \int_{\Omega} \nabla \mathbf{u} \cdot \nabla \mathbf{v}\, dx. \tag{44}$$

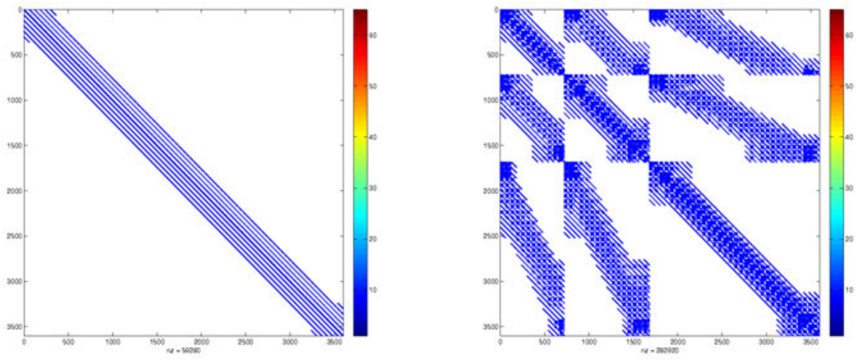


Fig. 1. Gram matrices of divergence-free scaling functions (left) and wavelets (right) built from Daubechies generators with $r = 3$, $j_{min} = 4$, $j = 6$

4.3 Right-Hand Side Computations

To solve system (38), we need to compute efficiently inner products $\langle \mathbf{u}, \Psi^{div}_{\mathbf{j},\mathbf{k}} \rangle$ and $\langle \mathbf{u}, \Psi^{\nabla}_{\mathbf{j},\mathbf{k}} \rangle$. This is achieved by using the decomposition of $\mathbf{u}$ in the wavelet bases in the suitable multiresolution analyses of $(L^2(\Omega))^d$ that contain alternatively the divergence-free or curl-free functions. To illustrate, we will explain the computation of $\langle \mathbf{u}, \Psi^{\nabla}_{\mathbf{j},\mathbf{k}} \rangle$ in the two dimensional case.

Let $(d^1_{\mathbf{j},\mathbf{k}})$ and $(d^2_{\mathbf{j},\mathbf{k}})$ denote respectively the coefficients of the decomposition of $\mathbf{u} = (\mathbf{u}_1, \mathbf{u}_2)$ on the wavelet basis of $(V^0_j \otimes V^D_j) \times (V^D_j \otimes V^0_j)$:

$$\mathbf{u}_1 = \sum_{\mathbf{j},\mathbf{k}} d^1_{\mathbf{j},\mathbf{k}} \; \psi^0_{j_1,k_1} \otimes \psi^D_{j_2,k_2} \qquad \mathbf{u}_2 = \sum_{\mathbf{j},\mathbf{k}} d^2_{\mathbf{j},\mathbf{k}} \; \psi^D_{j_1,k_1} \otimes \psi^0_{j_2,k_2}.$$

The computation of inner product $\langle \mathbf{u}, \Psi^{\nabla}_{\mathbf{j}',\mathbf{k}'} \rangle$ is

$$\begin{aligned}\langle \mathbf{u}, \Psi^{\nabla}_{\mathbf{j}',\mathbf{k}'} \rangle &= \sum_{\mathbf{j},\mathbf{k}} d^1_{\mathbf{j},\mathbf{k}} \; \langle \psi^0_{j_1,k_1} \otimes \psi^D_{j_2,k_2}, (\psi^D_{j'_1,k'_1})' \otimes \psi^D_{j'_2,k'_2} \rangle \\ &+ \sum_{\mathbf{j},\mathbf{k}} d^2_{\mathbf{j},\mathbf{k}} \; \langle \psi^D_{j_1,k_1} \otimes \psi^0_{j_2,k_2}, \psi^D_{j'_1,k'_1} \otimes (\psi^D_{j'_2,k'_2})' \rangle.\end{aligned}$$

In terms of coefficient matrices $[d^1_{\mathbf{j},\mathbf{k}}]$ and $[d^1_{\mathbf{j},\mathbf{k}}]$, it becomes

$$[\langle \mathbf{u}, \Psi^{\nabla}_{\mathbf{j}',\mathbf{k}'} \rangle] = \mathbf{C}^0_j \; [d^1_{\mathbf{j},\mathbf{k}}] \; \mathbf{M}_j + \mathbf{M}_j \; [d^2_{\mathbf{j},\mathbf{k}}] \; (\mathbf{C}^0_j)^t,$$

where $\mathbf{C}^0_j$ is the stiffness matrix of elements: $\langle \psi^0_{j,k}, (\psi^D_{j',k'})' \rangle$. The computation of the Gram and stiffness matrices $\mathbf{M}_j$, $\mathbf{C}^0_j$ is classical [4] (see also [17,20]).

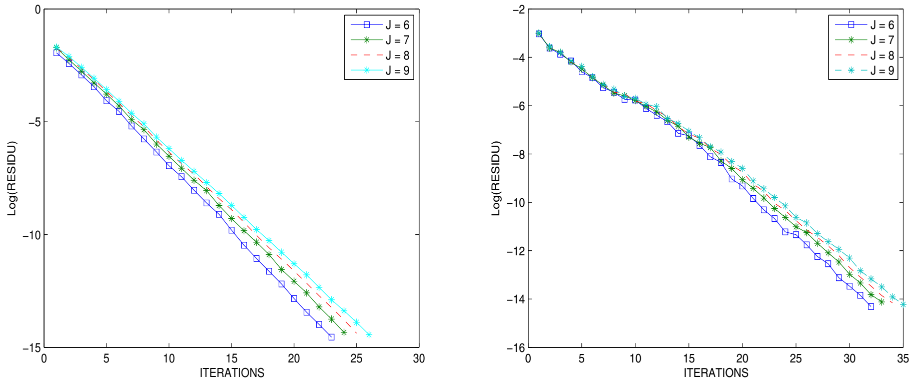


Fig. 2. Preconditioned conjugate gradient residual versus iteration number, for different values of the dimension index j: periodic case (left) and non periodic case (right). Daubechies generators ψ^1 with $r = 3$ vanishing moments, $j_{min} = 3$ in non periodic (2D case).

4.4 Divergence-Free and Curl-Free Gram Matrices Preconditioning

The tensorial decomposition (42) of $\mathbb{M}_{\text{curl}}$ is used to deduce a preconditioner from those of matrices $\mathbf{M}_j \otimes \mathbf{R}_j$ and $\mathbf{R}_j \otimes \mathbf{M}_j$. Let $\mathbf{I}_j$ be the identity matrix of dimension $(N_j - 2)$ and $\mathbf{I}_R$ be the diagonal matrix of $\mathbf{R}_j$:

$$[\mathbf{I}_j]_{k,k'} = \delta_{k,k'} \qquad \text{and} \qquad [\mathbf{I}_R]_{k,k'} = [\mathbf{R}_j]_{k,k'}\delta_{k,k'}, \quad 1 \le k, k' \le N_j - 2.$$

On one hand, as an optimal (and diagonal) preconditioner of $\mathbf{R}_j$ is given by the inverse of $\mathbf{I}_R$ (see [7,10]), readily we deduce optimal diagonal preconditioners of matrices $\mathbf{M}_j \otimes \mathbf{R}_j$ and $\mathbf{R}_j \otimes \mathbf{M}_j$ by respectively the inverse of matrices $\mathbf{I}_j \otimes \mathbf{I_R}$ and $\mathbf{I_R} \otimes \mathbf{I}_j$.

On the other hand, since $\mathbb{M}_{\text{curl}}$ is the 2D stiffness matrix of a scalar Laplacian on the basis $\{\psi^D_{j_1,k_1} \otimes \psi^D_{j_2,k_2}\}$, an optimal preconditioner is given by the inverse of its diagonal matrix [7,10]. Then, the inverse of the diagonal matrix $\mathbf{D}_j$ defined by

$$\mathbf{D}_j = \mathbf{I}_j \otimes \mathbf{I_R} + \mathbf{I_R} \otimes \mathbf{I}_j$$

is an optimal diagonal preconditioner for $\mathbb{M}_{\text{curl}}$.

However, the matrix $\mathbf{D}_j$ has the same size as $\mathbb{M}_{\text{curl}}$. To reduce the complexity, we replace the 2D matrix-vector product $\mathbf{D}_j(d^{\nabla}_{\mathbf{j},\mathbf{k}})$ by the following matrix-matrix products:

$$[\mathbf{D}_j(d^{\nabla}_{\mathbf{j},\mathbf{k}})] = [d^{\nabla}_{\mathbf{j},\mathbf{k}}]\mathbf{I_R} + \mathbf{I_R}[d^{\nabla}_{\mathbf{j},\mathbf{k}}]. \tag{45}$$

Because of the diagonal structure of $\mathbf{I_R}$, equation (45) is then reduced to term by term matrix product:

$$[\mathbf{D}_j(d^{\nabla}_{\mathbf{j},\mathbf{k}})]_{k,k'} = [d^{\nabla}_{\mathbf{j},\mathbf{k}}]_{k,k'} \cdot [\mathbf{I}^*_{\mathbf{R}}]_{k,k'} \quad \text{where} \quad [\mathbf{I}^*_{\mathbf{R}}]_{k,k'} = [\mathbf{I_R}]_{k,k} + [\mathbf{I_R}]_{k',k'}. \tag{46}$$

From equation ([46]), multiplying $(d^{\nabla}_{\mathbf{j},\mathbf{k}})$ by the matrix $\mathbf{D}_j^{-1}$ is therefore equivalent to divide term by term the matrix $[d^{\nabla}_{\mathbf{j},\mathbf{k}}]$ by $\mathbf{I}^*_{\mathbf{R}}$.

The preconditioner $\mathbf{I}^*_{\mathbf{R}}$ is also valid for the matrix $\mathbb{M}_{\text{div}}$ in dimension two $(d = 2)$ since $\mathbb{M}_{\text{div}} = \mathbb{M}_{\text{curl}}$.

The performance of the above preconditioner for $\mathbb{M}_{\text{curl}}$ was tested in two and three dimensions, using a preconditioned conjugate gradient method to solve system ([43]), with a random right hand side. Then we study the number of

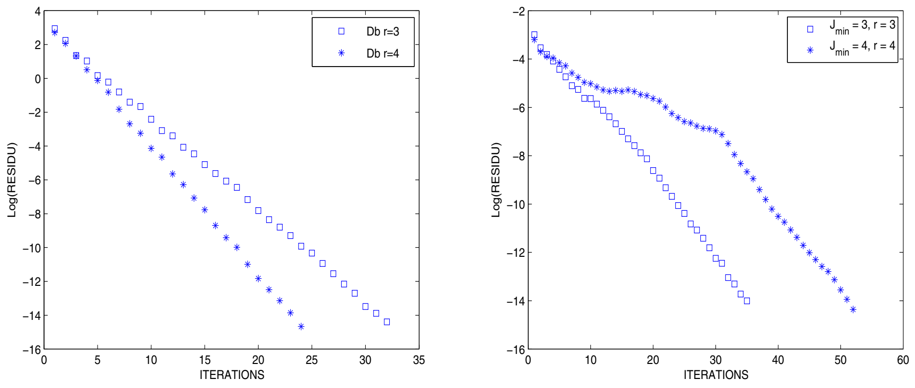


Fig. 3. Preconditioned conjugate gradient residuals versus iteration number, for different values of the approximation order r: periodic case (left) and non periodic case (right). Daubechies generators ψ^1 with $r = 3$ and $r = 4$ vanishing moments. The resolution is $j = 10$ in two dimension $(d = 2)$.

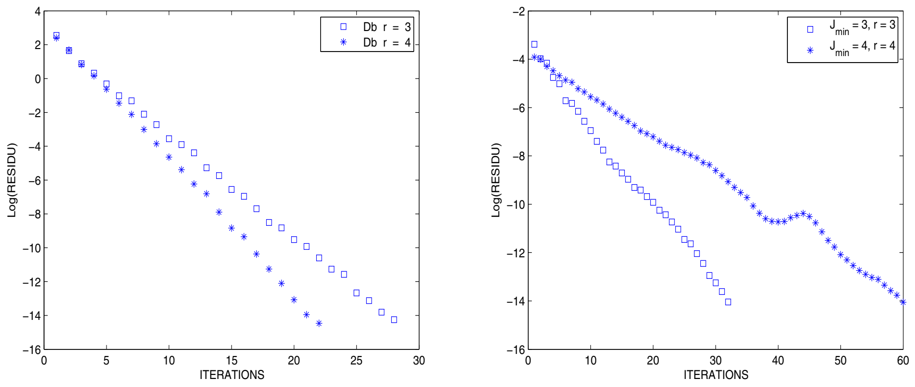


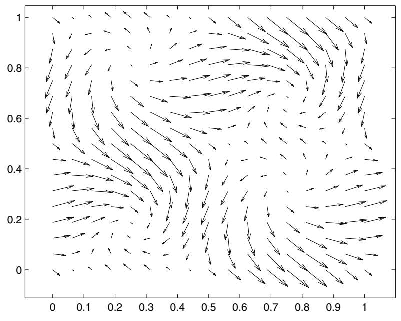
Fig. 4. Preconditioned conjugate gradient residuals versus iteration number, for different values of the approximation order r: periodic case (left) and non periodic case (right). Daubechies generators ψ^1 with $r = 3$ and $r = 4$ vanishing moments. The resolution is $j = 7$ in three dimension $(d = 3)$.

iterations needed to reach a given residual, first with respect to the dimension index j, second with respect to the regularity (approximation order r) of the basis functions. Figure 2 shows that the number of iterations does not increase significantly with the dimension index j, in the periodic and non periodic cases, which indicates that our preconditioner is quasi-optimal.

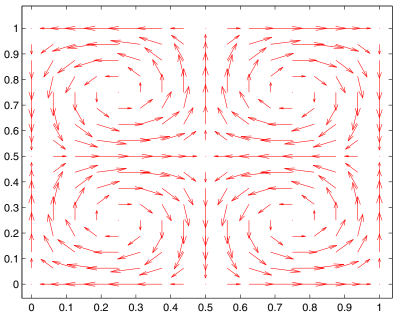
Figure 3 (two-dimensional case) and Figure 4 (three-dimensional case) highlight that the approximation order r speed up the convergence of the resolution. The behaviour of the slopes are less regular in the non periodic case, because of the influence of the smallest scale $j_{min} > 0$ (see [10] for similar conclusions).

4.5 Examples of Helmholtz-Hodge and Helmholtz Decomposition

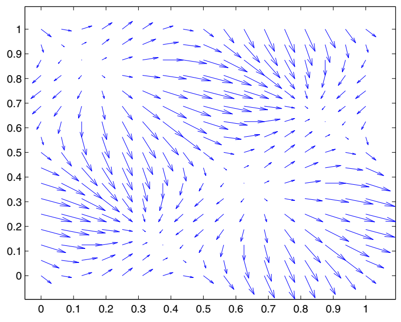
In this section, we carry out some experiments to illustrate and study the convergence rate of the Helmholtz-Hodge decomposition. First we show in dimension two, the Helmholtz decomposition of a vector field $\mathbf{u}$ (Figure 5), and its



(a) Initial vector field $\mathbf{u}^j_{2D}$

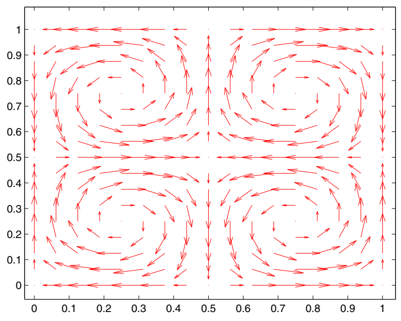


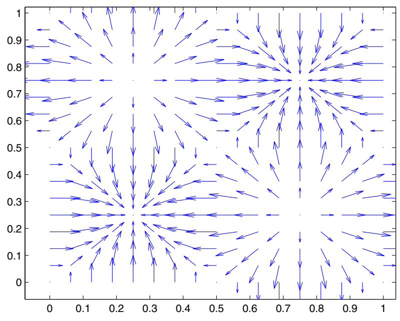
(b) Divergence-free part $\mathbf{u}^j_{\mathrm{div}}$



(c) Curl-free part $\mathbf{u}^j_{\mathrm{curl}} + \mathbf{u}^j_{\mathrm{har}}$

Fig. 5. Example of Helmholtz decomposition


(a) Divergence-free part $\mathbf{u}^j_{\mathrm{div}}$


(b) Curl-free part $\mathbf{u}^j_{\mathrm{curl}}$

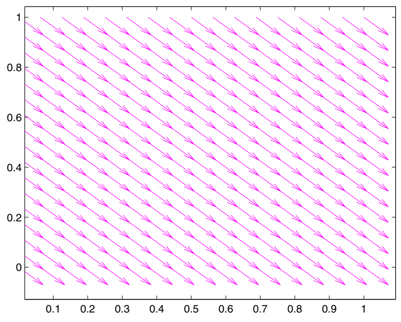

(c) Gradient of harmonic part $\mathbf{u}^j_{\mathrm{har}}$

Fig. 6. Example of Helmholtz-Hodge decomposition

Helmholtz-Hodge decomposition (Figure 6). The vector field $\mathbf{u}$ was constructed analytically:

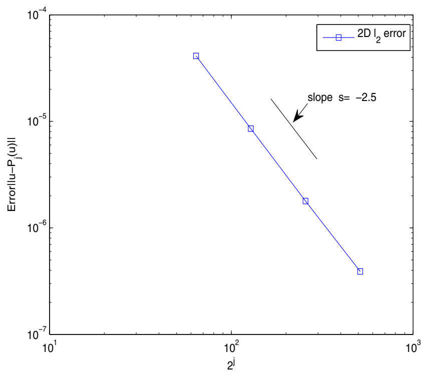
$$\mathbf{u}_{2D} = \mathbf{u}_{\mathrm{div}} + \mathbf{u}_{\mathrm{curl}} + \mathbf{u}_{\mathrm{har}},$$

where

$$\mathbf{u}_{\mathrm{div}} = \left| \begin{array}{l} \sin(2\pi x)^2 \sin(4\pi y) \\ -\sin(4\pi x)\sin(2\pi y)^2 \end{array} \right. , \ \mathbf{u}_{\mathrm{curl}} = \left| \begin{array}{l} \sin(4\pi x)\sin(2\pi y)^2 \\ \sin(2\pi x)^2 \sin(4\pi y) \end{array} \right. , \ \mathbf{u}_{\mathrm{har}} = (1/2, -1/4)$$

The terms of the decompositions are computed using the method described previously.

Then we investigate the convergence rate of the projection error onto the divergence-free vector space $\mathbf{V}^{div}_j$, in two and three dimensions. The tests have been performed on analytic fields, which we know the exact solutions. We used $\mathbf{u}_{2D}$ in two dimensions and $\mathbf{u}_{3D}$ in three dimensions:



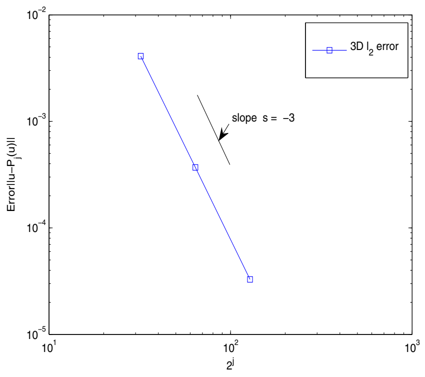


Fig. 7. ℓ_2-projection error onto $\mathbf{V}_j^{div}$ versus j. Two-dimensional case (left) and three-dimensional case (right). The generators $(\varphi^1, \tilde{\varphi}^1)$ correspond to biorthogonal splines with $r = \tilde{r} = 3$.

$$\mathbf{u}_{3D} = \left| \begin{array}{l} \sin(2\pi x)^2 \sin(4\pi y) \sin(4\pi z) + \sin(4\pi x) \sin(2\pi y)^2 \sin(2\pi z)^2 \\ \sin(4\pi x) \sin(2\pi y)^2 \sin(4\pi z) + \sin(2\pi x)^2 \sin(4\pi y) \sin(2\pi z)^2 \\ -2 \sin(4\pi x) \sin(4\pi y) \sin(2\pi z)^2 + \sin(2\pi x)^2 \sin(2\pi y)^2 \sin(4\pi z) \end{array} \right. \tag{47}$$

The solutions verify homogeneous Dirichlet boundary conditions by construction. Figure 7 plots the ℓ_2-projection errors in terms of the dimension index j with generators of approximation order $r = 3$. For both experiments (2D and 3D), the convergence rate follows the theoretical law of -2 predicted in ([11]).

5 Conclusion

In this paper, we have presented a practical algorithm to compute the Helmholtz-Hodge decomposition of a vector field in the hypercube. Our method is based on the existence of divergence-free and irrotational wavelet bases satisfying boundary conditions. After presenting the principles of their construction in any dimension, we have detailed the computation of each term of the decomposition, which requires the inversion of divergence-free and curl-free wavelet Gram matrices. We have used the tensorial structure of the bases to propose an optimal and diagonal preconditioning, to invert the system using a preconditioned conjugate gradient. Numerical tests on 2D and 3D analytical vector fields illustrate the potential of the approach, in terms of complexity and storage.

Since the Helmholtz-Hodge decomposition is a key ingredient for the analysis and simulation of incompressible flows, future works will present its application in numerical schemes for the Stokes and Navier-Stokes equations [17].

References

1. Amrouche, C., Bernardi, C., Dauge, M., Girault, V.: Vector potentials in three dimensional nonsmooth domains. Math. Meth. in the Applied Sciences 21, 823–864 (1998)
2. Amrouche, C., Seloula, N.: L^p-theory for vector potentials and Sobolev's inequalities for vector fields. C. R. Acad. Sci. Paris Ser. I 349, 529–534 (2011)
3. Battle, G., Federbush, P.: Divergence-free vector wavelets. Michigan Math. Journ. 40, 181–195 (1993)
4. Beylkin, G.: On the representation of operator in bases of compactly supported wavelets. SIAM J. Numer. Anal. 6, 1716–1740 (1992)
5. Chiavassa, G., Liandrat, J.: On the Effective Construction of Compactly Supported Wavelets Satisfying Homogeneous Boundary Conditions on the Interval. Appl. Comput. Harmon. Anal. 4, 62–73 (1997)
6. Chorin, A., Marsden, J.: A Mathematical Introduction to Fluid Mechanics. Springer, Heidelberg (1992)
7. Cohen, A.: Numerical Analysis of Wavelet Methods. Elsevier, Amsterdam (2003)
8. Cohen, A., Daubechies, I., Feauveau, J.-C.: Biorthogonal bases of compactly supported wavelets. Comm. Pure Appli. Maths. 45, 485–560 (1992)
9. Cohen, A., Daubechies, I., Vial, P.: Wavelets on the Interval and Fast Wavelet Transforms. Appl. Comput. Harmon. Anal. 1, 54–81 (1993)
10. Cohen, A., Masson, R.: Wavelet Methods for Second-Order Elliptic Problems, Preconditioning, and Adaptivity. SIAM J. on Sci. Comput. 21, 1006–1026 (1999)
11. Deriaz, E., Perrier, V.: Orthogonal Helmholtz decomposition in arbitrary dimension using divergence-free and curl-free wavelets. Appl. Comput. Harmon. Anal. 26, 249–269 (2009)
12. Deriaz, E., Perrier, V.: Divergence-free and curl-free wavelets in 2D and 3D, application to turbulent flows. J. of Turbulence 7, 1–37 (2006)
13. Dodu, F., Rabut, C.: Irrotational or Divergence-Free Interpolation. Numerisch Mathematic 98, 477–498 (2004)
14. Girault, V., Raviart, P.A.: Finite element methods for Navier-Stokes equations. Springer, Berlin (1986)
15. Grivet-Talocia, S., Tabacco, A.: Wavelet on the interval with optimal localization. Math. Models. Meth. Appl. Sci. 10, 441–462 (2000)
16. Jouini, A., Lemarié-Rieusset, P.G.: Analyse multirésolution biorthogonale sur l'intervalle et applications. Annales de l'I.H.P. Section C 10, 453–476 (1993)
17. Kadri-Harouna, S.: Ondelettes pour la prise en compte de conditions aux limites en turbulence incompressible. Phd thesis of Grenoble University (2010)
18. Kadri-Harouna, S., Perrier, V.: Divergence-free and Curl-free Wavelets on the Square for Numerical Simulation. Preprint hal-00558474 (2010)
19. Lemarié-Rieusset, P.G.: Analyses multi-résolutions non orthogonales, commutation entre projecteurs et dérivation et ondelettes vecteurs à divergence nulle. Revista Matemática Iberoamericana 8, 221–236 (1992)
20. Monasse, P., Perrier, V.: Orthogonal Wavelet Bases Adapted For Partial Differential Equations With Boundary Conditions. SIAM J. Math. Anal. 29, 1040–1065 (1998)
21. Petronetto, F., Paiva, A., Lage, M., Tavares, G., Lopes, H., Lewiner, T.: Meshless Helmholtz-Hodge decomposition. IEEE Trans. on Visu. and Comp. Graps. 16, 338–342 (2010)

22. Polthier, K., Preub, E.: Identifying Vector Field Singularities using a Discrete Hodge Decomposition. In: Hege, H.C., Polthier, K. (eds.) Visualization and Mathematics III, pp. 113–134. Springer, Heidelberg (2003)
23. Stevenson, R.: Divergence-free wavelet bases on the hypercube. Appl. Comput. Harmon. Anal. 30, 1–19 (2011)
24. Stevenson, R.: Divergence-free wavelet bases on the hypercube: Free-slip boundary conditions, and applications for solving the instationary Stokes equations. Math. Comp. 80, 1499–1523 (2011)
25. Tong, T., Lombeyda, S., Hirani, A.N., Desbrun, M.: Discrete multiscale vector field decomposition. ACM Transactions on Graphics (TOG) 22, 445–452 (2003)
26. Urban, K.: Wavelet Bases in H(*div*) and H(*curl*). Math. Comput. 70, 739–766 (2000)
27. Urban, K.: Wavelets in Numerical Simulation. Springer, Berlin (2002)

Finite Element Analysis with B-Splines: Weighted and Isogeometric Methods

Klaus Höllig, Jörg Hörner, and Axel Hoffacker

Universität Stuttgart, IMNG
Pfaffenwaldring 57, 70569 Stuttgart,
Germany

Abstract. Weighted and isogeometric methods use b-splines to construct bases for FEM. They combine the computational efficiency of regular grids with the geometric flexibility of CAD representations. We give a brief description of the key ideas of the two approaches, presenting them in a unified framework. In particular, we use b-spline nodes, to visualize the free parameters. Moreover, we explain how to combine features of both techniques by introducing weighted isogeometric finite elements. An error estimate for the resulting mixed method is given, and the performance of weighted approximations is illustrated by numerical examples.

Keywords: b-spline, finite element method, weight function, isogeometric approximation.

1 Introduction

B-splines play an important role in many areas of applied mathematics and engineering. With weighted[1] and isogeometric[2] methods, the advantages of the b-spline calculus are made available also to finite element techniques. Moreover, the new concepts provide a natural link from numerical simulation to geometric modeling where b-splines have long become a standard tool.

The two approaches are described in detail in the books *Finite Element Methods with B-Splines* [10] and *Isogeometric Analysis* [7], respectively. We also refer to [8,3,1,4] for a small sample of recent developments. In this paper, we give a brief introduction to some key ideas of both techniques, illustrating their basic features in the simplest possible setting. Moreover, we explain when a combination of both methods might be useful. We propose weighted isogeometric approximations which can, in particular, handle trim curves and surfaces efficiently. Of course, we would like to stimulate the interest of the reader to learn about all aspects of b-spline based finite elements, to implement algorithms for further applications, and to participate in the future development of the theory.

[1] Weighted extended b-splines (web-splines) were introduced by U. Reif, J. Wipper and the first author [12], cf. also http://www.web-spline.de

[2] Isogeometric Analysis was founded by T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs [14] .

We begin by reviewing basic facts about finite elements and b-splines (cf., e.g., [20,21,5,18]). In particular, we describe the concept of b-spline nodes, which is convenient for visualizing the degrees of freedom of numerical approximations. Then, weighted b-spline bases, isogeometric elements, and a new mixed method are discussed in turn. Moreover, we show that weighted isogeometric (mixed) elements approximate with optimal order. Finally, examples are presented which illustrate the performance of b-spline based simulations.

2 Finite Element Approximation

Many physical or engineering problems admit a variational formulation. This means that the function

$$x \mapsto u(x) \in \mathbb{R}$$

describing a phenomenon or process on a domain $D \subset \mathbb{R}^d$ minimizes an energy functional

$$u \mapsto \mathcal{Q}(u) = \int_D F(x, u, \nabla u, \ldots)\, dx$$

over a suitable Hilbert space H, which incorporates boundary conditions if necessary.

A finite element approximation

$$u_h = \sum_k c_k B_k$$

minimizes $\mathcal{Q}$ over a finite dimensional subspace $V_h = \operatorname{span}_k B_k$ of H:

$$\mathcal{Q}(u_h) = \min_{v_h \in V_h} \mathcal{Q}(v_h)\,,$$

where the discretization parameter h usually denotes a grid width. Clearly, the choice of V_h as well as of the basis functions or finite elements B_k is crucial for the accuracy and the efficiency of the resulting method. An enormous number of different possibilities is available – we will add several further choices in the next sections!

As a basic example, we consider Poisson's problem corresponding to

$$F = \frac{1}{2}\,|\nabla u|^2 - f(x)u\,,$$

with a given function f. If no boundary condition is imposed, the normal derivative of a solution u vanishes on the boundary ∂D. This so-called natural boundary condition does not have to be incorporated into the finite element subspace. A typical essential boundary condition is

$$u(x) = 0, \quad x \in \partial D\,. \tag{1}$$

It must be satisfied (at least approximately) by all elements of V_h, in particular by the basis functions B_k.

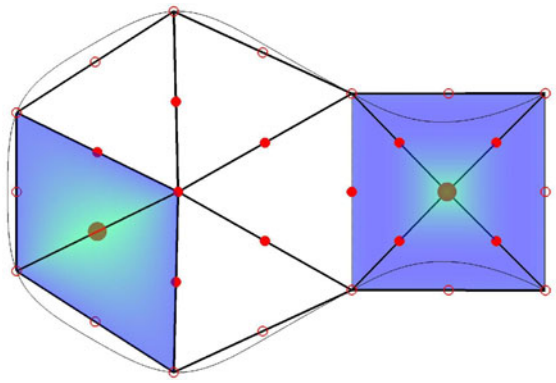


Fig. 1. Quadratic Lagrange elements on a triangulation

The standard classical finite element approximation of Poisson's problem on a two-dimensional domain D employs Lagrange elements on triangles. As is depicted in Figure 1, the degrees of freedom can be visualized by nodes at the positions of the interpolated values. To each node x^k corresponds a basis function B_k with $B_k(x^k) = 1$ and $B_k(x^\ell) = 0$ for $\ell \neq k$. The supports of two such finite elements are highlighted in the figure. The boundary condition (1) is imposed simply by assigning the value 0 to the boundary nodes (circles), leaving merely the values at the interior nodes (dots) as free parameters.

3 B-Splines

A (standard) d-variate b-spline b_k is a positive, bell-shaped, piecewise polynomial function of coordinate degree n. As is illustrated in Figure 2, smoothness and support are determined by the knots. The left figure visualizes the values of the b-spline by coloring the support on the grid. This style of graphic representation will be frequently used in the following. As shown on the right figure, cross sections of the graph coincide with scaled univariate b-splines.

It is convenient to associate a node x^k equal to the Greville abscissa with a b-spline b_k, i.e., x^k_ν is the average of the interior knots in the ν-th coordinate direction ($\nu = 1, \ldots, d$), counting multiplicities. The index $k = (k_1, ..., k_d)$ corresponds to the position of b_k on the grid separating the polynomial segments. Intuitively, the location of the node coincides with the point of strongest influence of the b-spline. The nodes will be used later on to visualize the degrees of freedom for spline approximations.

A spline is a linear combination of b-splines which have support on a grid-conforming hyper-rectangle R:

$$p = \sum_{k \sim R} c_k b_k \,,$$

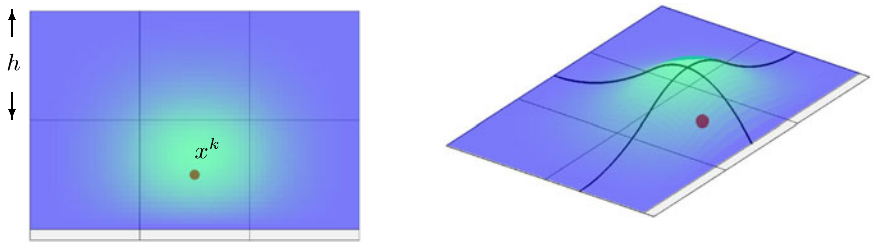


Fig. 2. Bi-quadratic b-spline with grid and node

where $k \sim R \Leftrightarrow k_\nu = 1, \ldots, m_\nu$ with $m_\nu + n + 1$ the number of knots in the ν-th coordinate direction. We can associate the free parameters or coefficients c_k with the b-spline nodes. Figure 3 shows two standard situations. Uniform splines (left) are spanned by translates of a single b-spline with obvious computational advantages. In particular, the node pattern is completely regular. For a boundary-conforming spline space (right), the boundary grid hyperplanes coincide with the boundary of the hyper-rectangle and have multiplicity $n+1$. This implies that the values of p along any of the hyper-rectangle boundaries are determined by the coefficients corresponding to the boundary nodes. For example, if all of these coefficients are 0, then p vanishes along the entire boundary of R.

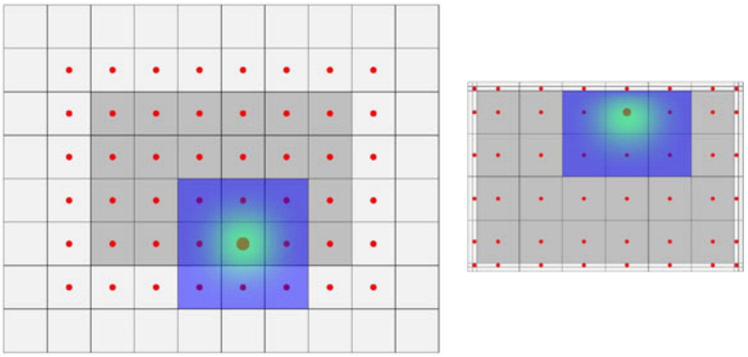


Fig. 3. Uniform and boundary-conforming bi-quadratic spline spaces

If the coefficients of a spline Φ are points $C_k \in \mathbb{R}^d$, then

$$\xi \mapsto x = \Phi(\xi) = \sum_{k \sim R} C_k b_k(\xi), \quad \xi \in R,$$

describes a transformation of the parameter hyper-rectangle R. The control net formed by the array of points C_k provides a qualitative description of the

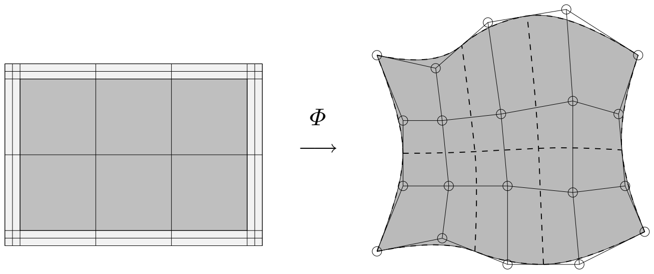


Fig. 4. Bi-quadratic b-spline parametrization with control net and grid

deformation caused by Φ, as does the isoparametric grid (image of the partition of the spline space under Φ).

Usually, boundary-conforming spline spaces are used for modeling parameter transformations as is the case for the example in Figure 4. The advantage is that the boundary of the image is determined entirely by the points C_k corresponding to the boundary nodes.

4 Weighted B-Splines

In order to approximate a function u on a domain D we can simply use splines defined on a hyper-rectangle R containing D:

$$u \approx u_h = \sum_{k \sim R} c_k b_k \, .$$

To emphasize that only those b-splines with some support in D are relevant, we set irrelevant coefficients to 0. In the example of Figure 5, uniform bi-quadratic b-splines are used; the solid relevant nodes correspond to the free parameters c_k, $k \sim D$.

Perhaps somewhat surprisingly, the simple procedure works well for unconstrained variational problems. Just restricting the b-splines to the simulation region D, provides very accurate finite element approximations u_h for problems with natural boundary conditions.

To incorporate essential boundary conditions, we resort to an idea already proposed by Kantorovich and Krylov [15]. We represent the domain D in implicit form via a weight function w,

$$D : w > 0 \, ,$$

as illustrated on the left of Figure 5. Multiplying the b-splines b_k by w, we obtain a suitable finite element basis for constrained problems:

$$B_k = w b_k , \quad k \sim D \, . \tag{2}$$

By construction, any of the weighted elements satisfy $w(x)b_k(x) = 0$, $x \in \partial D$. The precise adaptation to the boundary is apparent from the sample elements highlighted on the right of the figure.

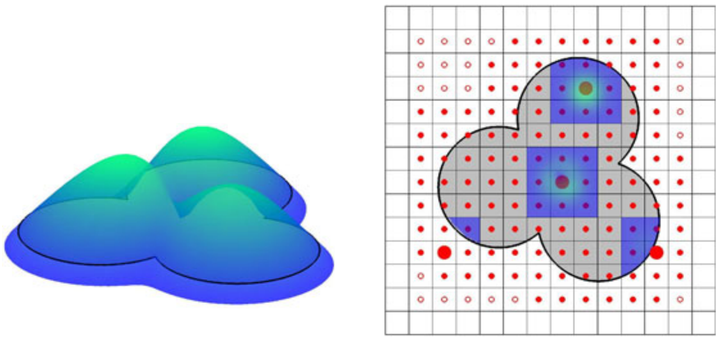


Fig. 5. Weight function and weighted bi-quadratic b-splines with grid and nodes

Weight functions can be constructed in various ways. For example, the smoothed distance to the boundary provides a general purpose solution. An elegant procedure is Rvachev's R-function method [17]. It combines elementary weight functions according to Boolean operations and is thus particularly well suited for simulations in conjunction with constructive solid geometry. The weight function on the left of Figure 5 was constructed in this fashion.

The weighted basis functions wb_k share all properties of standard finite elements, except for stability. For b-splines b_j, which do not have at least one of the grid cells of their support in D, the norm of wb_j is very small. For moderate grid widths this does not present any problems. In fact, the weighted basis (2) usually provides adequate approximations. However, for certain algorithms stability can be crucial, as $h \to 0$. To obtain stable finite elements, we combine neighboring b-splines with support near the boundary, forming so-called weighted extended b-splines (web-splines), introduced by U. Reif, J. Wipper, and the first author in [12]:

$$B_i = \sum_k e_{i,k}(wb_k), \quad i \in I\,.$$

The set I comprises all indices of inner b-splines, i.e., those b_i with at least one grid cell of their support in D.

The mathematics leading to the proper choice of the extension coefficients $e_{i,k}$ is somewhat subtle. However, the basis change $wb_k \to B_i$ can be implemented efficiently. In effect, the work amount is comparable to a sparse preconditioning procedure, since the (generalized) matrix $(e_{i,k})_{i\in I,k\sim D}$ has few off-diagonal entries.

In two variables, this construction is not even necessary. B. Mößner and U. Reif have made the surprising discovery that b-splines can be stabilized simply by scaling [16]. This is in agreement with many of our numerical experiments which indicated that stabilization can often be omitted. The scaling, proposed by B. Mößner and U. Reif, is inherent to most preconditioners for iterative solvers.

We reviewed in this section just the basic idea of the web-method, providing the prerequisites for the new mixed method, described in Section 6. A more detailed introduction can be found on the web-site `www.web-spline.de` which also provides examples and further references.

5 Isogeometric Elements

Often it is possible to decompose a domain into simple patches which can be described as images of hyper-rectangles:

$$D = \bigcup_\alpha \Phi_\alpha(R_\alpha)\,.$$

In fact, CAD descriptions in solid geometry provide spline parametrizations of the form described in Section 3. A generalization of the classical isoparametric concept suggests itself. Isogeometric analysis, founded by T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs, provides a natural link from CAD models to FEM simulations. We briefly scetch the main idea of this powerful technique in a very simple setting, just providing sufficient detail to introduce a possible combination with weighted methods in the following section. For a comprehensive description of isogeometric analysis, we refer to [7].

As is illustrated in Figure 6, we can transform boundary-conforming b-splines defined on each of the hyper-rectangles R_α, with the aid of the mappings Φ_α. In other words, we use the basis functions

$$D \ni x \mapsto B_{k,\alpha}(x) = b_{k,\alpha}(\xi), \quad \xi = \Phi_\alpha^{-1}(x)\,,$$

the so-called isogeometric b-splines. Clearly, to ensure continuity, consistency at patch boundaries is crucial. This means that the restrictions of b-splines to a common patch boundary must coincide (nodes connected by red lines in the figure) and share the same coefficient.

We visualize the degrees of freedom by transforming the nodes associated with the b-splines to D,

$$\xi^{k,\alpha} \mapsto x^{k,\alpha} = \Phi_\alpha(\xi^{k,\alpha})\,,$$

as is illustrated in Figure 6. In particular, in view of consistency, nodes $x^{k,\alpha}$ on a common patch boundary are shared by the b-splines of the neighboring patches. Moreover, if essential boundary conditions are imposed as in the example in the figure, coefficients associated with nodes on outer boundaries (marked by circles) are set to 0.

It is convenient to also use boundary-conforming b-splines to represent the parametrizations Φ_α. Typically, the grid for the isogeometric elements then is a refinement of the grid for the parametrization. The degrees do not have to match. In the example in Figure 6, bi-quadratic b-splines are used throughout, which is, of course, a slight computational advantage. The basis functions $B_{k,\alpha}$ are slight perturbations of standard b-splines adapting to the grid determined by the mappings Φ_α.

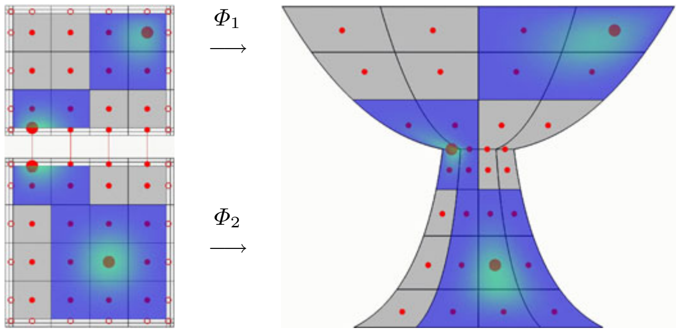


Fig. 6. Domain parametrization and bi-quadratic isogeometric b-splines with grid and nodes

Despite the nonlinear transformations involved, isogeometric methods can be implemented efficiently. Finite element integrals of the form

$$\int_D F(x, B(x), \nabla B(x), \ldots)\, dx, \quad B(x) = b(\xi),\, \xi = \Phi^{-1}(x)\,,$$

are computed over the relevant parameter hyper-rectangle R. By the chain rule and the formula for changing integration variables, the integral equals

$$\int_R F(\Phi(\xi), b(\xi), \nabla b(\xi)(J\Phi(\xi))^{-1}, \ldots)\; |\mathrm{det} J\Phi(\xi)|\; d\xi\,, \tag{3}$$

where $J\Phi$ denotes the Jacobi matrix of the transformation. Hence, matrix assembly does not require inverting the transformations of the parameter hyper-rectangles, a key feature familiar from classical isoparametric methods.

6 Weighted Isogeometric Approximation

Some commonly used CAD representations employ patches parametrized over trimmed parameter hyper-rectangles. A simple example is shown in Figure 7. To apply the standard isogeometric method, the image domain would have to be partitioned into deformed hyper-rectangles. As is already apparent from the elementary shape in the figure, it is not always easy to find a natural partition, in particular with few, only moderately distorted patches.

A possible remedy is a combination of the weighted and isogeometric approaches described, for the sake of simplicity, only for a single patch. We use weight functions to represent the trim curves or surfaces. The constraints can be specified either in the parameter hyper-rectangle R or the physical domain D. If both variants are used as in the example of Figure 7, the active portion R_a of the parameter domain consists of the points $\xi = \Phi^{-1}(x)$ with

$$w_R(\xi) > 0\,, \quad w_D(x) > 0$$

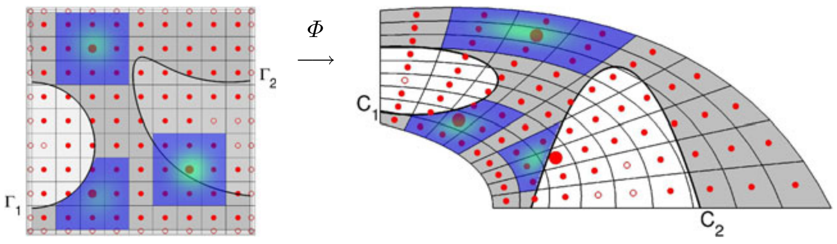


Fig. 7. Trimmed parameter rectangle and domain with bi-quadratic weighted isogeometric b-splines

$(\Gamma_1 : w_R = 0,\, C_2 : w_D = 0)$. Forming products with the boundary-conforming b-splines, we obtain the weighted isogeometric b-splines

$$x \mapsto B_k(x) = w_R(\xi) w_D(x) b_k(\xi), \quad x = \Phi(\xi)\,.$$

These basis functions are suited for problems with essential boundary conditions on the image of the trim curves or surfaces. If essential boundary conditions are also prescribed on the outer boundary of D, then the coefficients associated with the boundary nodes are set to 0. In Figure 7, the degrees of freedom are visualized in the usual way. The nodes $\xi^k \in R$ are transformed to the physical domain via $\Phi :\ x^k = \Phi(\xi^k)$.

Several weighted isogeometric b-splines B_k are highlighted on the right of Figure 7. As in the previous examples, degree 2 was used for the parametrization of the domain as well as for the b-spline basis. The weight functions coincide with the standard implicit representations of a circle and a parabola.

Trimming will usually lead to weighted isogeometric b-splines with small support within the domain D. To avoid the resulting instabilities, the stabilization measures, which were briefly mentioned at the end of Section 4, can be applied: simple scaling in two and extension in three variables [16,12]. This yields a stable basis if Φ and Φ^{-1} are smooth. It seems, however, that stabilization can be omitted in many cases. We have found (cf. the example at the end of the next section and the remark in connection with the first example in Section 8) that the accuracy of approximations is affected by instability only for extremely small grid widths.

Any weighted approximation requires special integration routines for boundary cells. This is straightforward in two dimensions (cf. [10], Section 8.4) for any degree and in three dimensions for degree 1, where an interesting preprocessing technique can be used (cf. [13]). Routines for three dimensional integration over cell intersections with general NURBS-domains have been developed (cf. the MIND project: `www.imng.uni-stuttgart.de/LstNumGeoMod/Hoerner/mind/`), and perform sufficiently well for smooth boundary portions. For complicated intersection patterns, as produced by curved edges and corners, integration can be time consuming. Fortunately, asymptotically (for small grid width), the percentage of these cases becomes small.

7 Error Estimate

We show in this section that weighted isogeometric approximations have, in general, the optimal approximation order for standard elliptic problems with smooth solutions. As a typical case, we consider Poisson's equation

$$-\Delta u = f$$

with mixed boundary conditions on a bounded domain $D \subset \mathbb{R}^d$ of the form shown in Figure 8 for $d = 2$. On the inner boundary C or trim surface, which encloses a simply connected subdomain, homogeneous Dirichlet boundary conditions are prescribed ($u_{|C} = 0$) and, on the outer boundary $\partial D \backslash C$, the solution satisfies the Neumann condition $\partial_\perp u = 0$ (vanishing normal derivative).

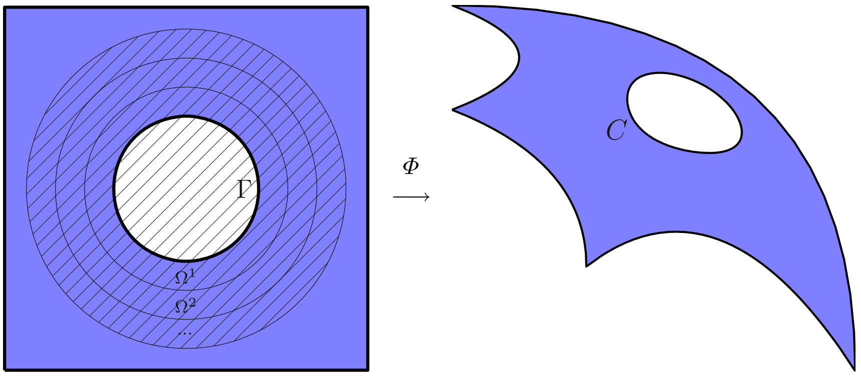


Fig. 8. Regularly parametrized domain $D = \Phi(\Omega)$, $\Omega \subset R$, with a smooth inner boundary $C = \Phi(\Gamma)$

We assume that the untrimmed domain is the image of a hyper-rectangle R under an $(n + 1)$ times continuously differentiable bijective transformation Φ with nonsingular Jacobian and denote by $\Omega = \Phi^{-1}(D)$ the trimmed parameter domain. Moreover, the trim surface C should be smooth, i.e., representable in implicit form via a smooth weight function w_D (or via $w_R = w_D \circ \Phi$ on the parameter hyper-rectangle R) with $w_D = 0 \wedge \operatorname{grad} w_D \neq 0$ on C and $w_D > 0$ in D and on the outer boundary. Under these hypotheses, the following error estimate holds.

Theorem (Error of Weighted Isogeometric Finite Elements**).** *The weighted isogeometric Ritz-Galerkin approximation u_h of degree $\le n$ to a solution $u \in H^{n+1}(D)$ of the mixed Poisson problem described above satisfies*

$$\|u - u_h\|_{1,D} \le \operatorname{const}(D, w_D, \Phi, n)\, h^n\, \|u\|_{n+1,D}\,,$$

where $\|\ \|_{\ell,D}$ denotes the norm for the Sobolev space $H^\ell(D)$ of functions with square integrable ℓ-th order partial derivatives.

Proof. To simplify notation we use the symbols $\preceq, \asymp$ for inequalities with generic constants in one and both directions, which may depend on D, w_D, Φ, and n but neither on u nor on h. Moreover, we use a tilde for functions defined on the trimmed parameter hyper-rectangle $\Omega = \Phi^{-1}(D)$:

$$u(x) = \tilde{u}(\xi),\, u_h(x) = \tilde{u}_h(\xi),\, \ldots, \quad x = \Phi(\xi),\, \xi \in \Omega \subset R\,,$$

etc.

The proof relies on results and techniques from isogeometric analysis [2,7] and the theory of weighted approximations [12,10]; after all, the theorem pertains to a combination of key features of the two approaches. Moreover, the (by now!) standard error estimates for splines (c.f. the classical books by C. de Boor and L.L. Schumaker [5,18]) are crucial for our arguments. We refer also to [9], where a weaker version of the theorem was obtained, for some of the preliminary arguments.

We begin by noting that the composition with Φ or Φ^{-1} and multiplication by a smooth weight function w are bounded operations with respect to Sobolev norms:

$$\|p\|_{\ell,D} \preceq \|\tilde{p}\|_{\ell,\Omega} \preceq \|p\|_{\ell,D}, \quad \|wp\|_{\ell,D} \preceq \|p\|_{\ell,D}\,. \tag{S}$$

This elementary observation follows from the chain rule, the formula for transformation of multiple integrals, Leibniz' rule, and the fact that Φ, Φ^{-1}, and w are sufficiently smooth.

Now we observe that, by Cea's Lemma, the error of u_h can be bounded, up to a constant factor, by the error of the best approximation from the finite element subspace. Hence, it suffices to construct a linear combination

$$w_D v_h = \sum_k c_k\, w_D(b_k \circ \Phi^{-1})$$

of weighted isogeometric finite elements $B_k = w_D(b_k \circ \Phi^{-1})$, which approximates $u \in H^{n+1}(D)$ with the desired order:

$$\|u - u_h\|_{1,D} \preceq \|u - w_D v_h\|_{1,D}\,,$$

as just noted in the preceeding sentence.

By the first inequalities in (S), the change of variables induced by the parametrization Φ is bounded with respect to Sobolev norms. In particular,

$$\|u - w_D v_h\|_{1,D} \preceq_{(S)} \|\tilde{u} - w_R \tilde{v}_h\|_{1,\Omega}, \quad \|\tilde{u}\|_{n+1,\Omega} \preceq_{(S)} \|u\|_{n+1,D}\,.$$

Hence, we may construct the approximation on the parameter hyper-rectangle R.

An appropriate linear combination of weighted b-splines

$$w_R \tilde{v}_h = \sum_k c_k\, w_R b_k \approx \tilde{u}$$

($w_R\tilde{v}_h = (w_D v_h)\circ\Phi$, $w_R b_k = B_k\circ\Phi$) is simply obtained by choosing for $\tilde{v}_h = \sum_k c_k b_k$ a quasi-interpolant of

$$\tilde{v} = \tilde{u}/w_R\,.$$

Here, we use an extension of $\tilde{v}$ to all of $\mathbb{R}^d$ (also denoted by $\tilde{v}$) to avoid technical difficulties in the construction of quasi-interpolant functionals near the boundaries of Ω. The existence of bounded extensions with respect to Sobolev norms,

$$\|\tilde{v}\|_{\ell,\mathbb{R}^d} \preceq \|\tilde{v}\|_{\ell,\Omega}\,, \tag{E}$$

was established by Calderon and Stein [6,19].

Summarizing, after these preliminaries, to prove the theorem we have to show that

$$\|w_R\tilde{v} - w_R\tilde{v}_h\|_{1,\Omega} \preceq h^n\|\tilde{u}\|_{n+1,\Omega}, \quad w_R\tilde{v} = \tilde{u}\,. \tag{A}$$

It seems as if we are done (cf. also the remark after the proof) since multiplication by w_R is a bounded operation and the quasi-interpolant $\tilde{v}_h \approx \tilde{v}$ approximates smooth functions in the H^1-norm with order $O(h^n)$. This is indeed the case for pure one-patch isogeometric approximations (no weighting, $w_R \equiv 1$, $\tilde{v} = \tilde{u}$) in a very special setting. The main difficulty we are facing in the presence of essential boundary conditions on a trim surface is that $\tilde{u} \in H^{n+1}(\Omega)$ does not imply $\tilde{v} \in H^{n+1}(\Omega)$, i.e., $\tilde{v}$ is not sufficiently regular to yield the maximal quasi-interpolation order. The division by w_R in the definition of $\tilde{v}$ causes the loss of roughly one order of differentiation as is already apparent from univariate examples. This problem is overcome with techniques developed in [12] (cf. also [10], Section 5.5), in particular with two fundamental inequalities, which we restate for convenience of the reader in a form appropriate for the mixed boundary value problem under consideration.

For any subdomain $U \subset \Omega$ with distance $\delta > 0$ to the inner boundary $\Gamma = \Phi^{-1}(C)$, the functions $\tilde{v}$ and $\tilde{u} = w_R\tilde{v}$ satisfy

$$\|\tilde{v}\|_{n+1,U} \preceq \delta^{-1}\left(\|\tilde{u}\|_{n+1,U} + \|\tilde{v}\|_{n,U}\right)\,. \tag{R1}$$

Moreover,

$$\|\tilde{v}\|_{n,\Omega} \preceq \|\tilde{u}\|_{n+1,\Omega}\,. \tag{R2}$$

In the references cited, the estimates were given for a smooth weight function which vanishes to first order on the entire boundary (no Neumann part). They also apply in the present context since, in a neighborhood Ω' of the Neumann boundary ∂R, the weight function w_R is bounded from below by a positive constant. This implies that $\|\tilde{v}\|_{n+1,\Omega'} \preceq_{(S)} \|\tilde{u}\|_{n+1,\Omega'}$, i.e., near the Neumann boundary both estimates, which essentially pertain to boundary behavior, are trivial.

Proceeding with the estimate of the error of the quasi-interpolant $\tilde{v}_h \approx \tilde{v}$ (assertion (A)), we have to take the two different types of boundary conditions, Dirichlet and Neumann, into account; the inner and outer boundary have to be treated in a slightly different fashion. To this end, we split the function $\tilde{v}$ to be approximated with the aid of a partition of unity into two parts:

$$\tilde{v} = \tilde{v}^D + \tilde{v}^N .$$

As is indicated by the superscripts, the support Ω^D of $\tilde{v}^D$ contains a neighborhood of the Dirichlet boundary Γ and the Neumann boundary ∂R is covered by $\operatorname{supp} \tilde{v}^N = \Omega^N$. More precisely, we choose $\Omega^D \subset R$ with positive distance from ∂R and $\Omega^N \subset (\Omega \cup {}^c R)$ with positive distance from Γ (cf. Figure 8). Moreover, the Sobolev norms of $\tilde{v}^D$ and $\tilde{v}^N$ are bounded in terms of the corresponding Sobolev norms of $\tilde{v}$. To accomplish this splitting, we choose two smooth nonnegtive functions (e.g., linear combinations of b-splines!) χ^D and χ^N with the appropriate supports and

$$\chi^D(\xi) + \chi^N(\xi) = 1, \quad \xi \in \mathbb{R}^d .$$

Then we set $\tilde{v}^D = \chi^D \tilde{v}$, $\tilde{v}^N = \chi^N \tilde{v}$. Clearly, by linearity, we have the same decomposition for the quasi-interpolants, i.e., $\tilde{v}_h = \tilde{v}_h^D + \tilde{v}_h^N$ ($\tilde{v}_h^*$ is a quasi-interpolant of $\chi^* \tilde{v}$).

As a final preparation for the main argument, we recall a standard estimate for quasi-interpolants $\tilde{p}_h$ with uniform b-splines b_k of functions $\tilde{p}$, defined on $\mathbb{R}^d$:

$$\|\tilde{p} - \tilde{p}_h\|_{m,U} \preceq h^{n+\nu-m} \|\tilde{p}\|_{n+\nu,U_h}, \quad 0 \le m \le n,\ \nu = 0, 1 , \tag{Q}$$

where U_h denotes the union of all b-spline supports overlapping the set U (actually, $\nu = m - n, \ldots, -1$ is possible; but not needed here). There exist many constructions for quasi-interpolants $\tilde{p}_h$; any of them which has the above approximation property is adequate for our purposes.

Referring to assertion (A), we now estimate each part (Neumann and Dirichlet) of the error

$$w_R \tilde{v} - w_R \tilde{v}_h = (w_R \tilde{v}^N - w_R \tilde{v}_h^N) + (w_R \tilde{v}^D - w_R \tilde{v}_h^D)$$

$(w_R \tilde{v} = \tilde{u})$ in turn.

The estimate of the error for the Neumann part is straightforward:

$$\begin{aligned} \|w_R \tilde{v}^N - w_R \tilde{v}_h^N\|_{1,\Omega} &\preceq_{(S)} \|\tilde{v}^N - \tilde{v}_h^N\|_{1,\Omega_h^N} \preceq_{(Q)} h^n \, \|\tilde{v}^N\|_{n+1,\Omega^N} \\ &\preceq_{(E)} h^n \, \|\tilde{v}^N\|_{n+1,\Omega} \preceq_{(S)} h^n \, \|\tilde{u}\|_{n+1,\Omega} ; \end{aligned} \tag{N}$$

the first inequality because $\operatorname{supp}(\tilde{v}^N - \tilde{v}_h^N) \subseteq \Omega_h^N$, the second inequality because $\operatorname{supp} \tilde{v}^N \subseteq \Omega^N$, the last inequality because $\tilde{v}^N = \chi^N \tilde{v} = \chi^N \tilde{u}/w_R$, χ^N is smooth, and $w_R \ge c > 0$ on $\Omega \cap \Omega^N = \Omega \cap \operatorname{supp} \chi^N$, noting that this set has a positive distance from Γ.

For analyzing the Dirichlet part in the decomposition of the error $w_R \tilde{v} - w_R \tilde{v}_h$, we introduce the abbreviation

$$w_R \tilde{v}^D - w_R \tilde{v}_h^D = w_R \tilde{e}_h ,$$

i.e., $\tilde{e}_h$ is the error of the quasi-interpolant of $\tilde{v}^D$. To estimate the H^1-norm of $w_R\tilde{e}_h$, we have to take the weight function more explicitly into account. We use the inequality

$$\|w_R\tilde{e}_h\|_{1,U} \preceq \sup_{\xi\in U} |w_R(\xi)| \; \|\tilde{e}_h\|_{1,U} + \|\tilde{e}_h\|_{0,U} \tag{I}$$

which follows directly from the definition of the H^1- and L_2-norm and is valid for any subset U of Ω. Indeed, with $\| \; \|_0$ denoting the L_2-norm and by ∂_ν the partial derivative with respect to the ν-th variable,

$$\begin{aligned}\|we\|_{1,U}^2 &= \|we\|_{0,U}^2 + \sum_\nu \|\partial_\nu(we)\|_{0,U}^2 \\ &\preceq_{(S)} \{ \, \|e\|_{0,U}^2 + \sum_\nu \sup_U |\partial_\nu w|^2 \, \|e\|_{0,U}^2 \, \} + \sum_\nu \sup_U |w|^2 \, \|\partial_\nu e\|_{0,U}^2 \, ,\end{aligned}$$

where $\{\ldots\} \preceq_{(S)} \|e\|_{0,U}^2$ and the last term is $\le \sup_U |w|^2 \, \|e\|_{1,U}^2$.

To capture the interplay between the smallness of w and the lack of regularity of $\tilde{v}$ near Γ, we cover $\Omega \cap \Omega_h^D$ (recall that Ω^D contains Γ and has a positive distance from the Neumann boundary ∂R) by strips Ω^1, Ω^2, ... with widths proportional to h and at least twice as large as the diameter of a b-spline support (cf. Figure 8). Using the inequality (I), we estimate $w_R\tilde{e}_h$ on each of these strips in turn. To this end the two terms on the right of the inequality are bounded with the aid of (Q), taking also the size of the supremum into account.

<u>strip Ω^1</u>: Since w_R is smooth and vanishes on Γ, we have $w_R(\xi) \preceq h$ on Ω^1. This implies

$$\begin{aligned}\|w_R\tilde{e}_h\|_{1,\Omega^1} &\preceq_{(I,Q)} h\, h^{n-1}\|\tilde{v}^D\|_{n,\Omega_h^1} + h^n\|\tilde{v}^D\|_{n,\Omega_h^1} \\ &\preceq_{(E,S)} h^n\|\tilde{v}\|_{n,\Omega} \preceq_{(R2)} h^n\|\tilde{u}\|_{n+1,\Omega} \, .\end{aligned} \tag{D1}$$

Here, (Q) was used with $m = 1, \nu = 0$ (first term) and $m = 0, \nu = 0$ (second term). We see that the inferior order for the H^1-norm of the error $\tilde{e}_h = \tilde{v}^D - \tilde{v}_h^D$ of the quasi-interpolant (first term) is compensated by the fact that w_R is small near the inner boundary Γ.

<u>strips Ω^ℓ, $\ell > 1$</u>: By construction,

$$\delta = \operatorname{dist}(\Gamma, \Omega_h^\ell) \asymp \operatorname{dist}(\Gamma, \Omega^\ell) \asymp \ell h \, .$$

To this end we note that the width of Ω^1 is at least twice as large as a b-spline support, so that this assertion is valid in particular for Ω_h^2; the enlarged set does not touch Γ. Moreover, by our assumptions on the weight function, $w_R \preceq \ell h$ on Ω_h^ℓ. Hence, since $\delta^{-1} \preceq (\ell h)^{-1}$,

$$\begin{aligned}\|w_R\tilde{e}_h\|_{1,\Omega^\ell} &\preceq_{(I,Q)} (\ell h)\, h^n\|\tilde{v}^D\|_{n+1,\Omega_h^\ell} + h^{n+1}\|\tilde{v}^D\|_{n+1,\Omega_h^\ell} \\ &\preceq_{(S)} \ell h^{n+1}\|\tilde{v}\|_{n+1,\Omega_h^\ell} \preceq_{(R1)} h^n \left(\|\tilde{u}\|_{n+1,\Omega_h^\ell} + \|\tilde{v}\|_{n,\Omega_h^\ell} \right) .\end{aligned}$$

Here, (Q) was used with $m = 1, \nu = 1$ (first term) and $m = 0, \nu = 1$ (second term). The fact that, for $\ell > 1$, Ω^ℓ and Ω_h^ℓ have a positive distance from Γ is crucial. This is the reason why the case $\ell = 1$ has to be treated differently.

Squaring the inequality, noting that $(\alpha + \beta)^2 \preceq \alpha^2 + \beta^2$, and summing over $\ell > 1$, we obtain the error bound also on the remaining part of the support Ω_h^D of $\tilde{e}_h$ within Ω:

$$\|w_R \tilde{e}_h\|^2_{1,\cup_{\ell>1}\Omega^\ell} \preceq h^{2n} \left(\|\tilde{u}\|^2_{n+1,\Omega} + \|\tilde{v}\|^2_{n,\Omega}\right) \preceq_{(R2)} h^{2n} \|\tilde{u}\|^2_{n+1,\Omega} , \qquad \text{(D2)}$$

noting that $\Omega_h^\ell \subset \Omega$ for small enough h and these enlarged strips overlap at most twice; $\Omega_h^\ell \cap \Omega_h^{\ell'} = \emptyset$ for $|\ell - \ell'| > 1$. Since

$$\Omega \cap \operatorname{supp} \tilde{e}_h = \Omega \cap \Omega_h^D \subset \cup_{\ell \geq 1} \Omega^\ell ,$$

combining the estimates (D1,D2) yields the desired bound also for the Dirichlet part of the error. Together with the estimate (N) this proves (A) and thereby the theorem. □

The arguments have been somewhat technical. This reflects the complexity of the approximation process which involves b-splines of arbitrary degree, curved boundaries, parametrizations, and weight functions. However, we note that a substantially simpler argumentation is possible if one is content with an error estimate of the form

$$\|u - u_h\|_{1,D} = O(h^n) ,$$

which does not show the precise dependence on the regularity of u. A result of this type neither requires the splitting of the error nor the regularity results (E), (R1), and (R2). With the aid of (Q) a relatively short proof is possible if one assumes, e.g., that u has continuous partial derivatives up to order $n + 2$. While we think that the above estimate is adequate for many purposes, usually, in finite element analysis, bounds with optimal regularity are preferred. They are essential for certain applications, e.g., the convergence analysis for multigrid algorithms, and certainly more appealing from an aesthetic point of view.

It is clear, that we can obtain analogous estimates for any standard elliptic second order problem with smooth solutions and smooth Dirichlet boundaries. More delicate (not done also for pure weighted approximations) is the analysis for problems with singularities due to reentrant corners or discontinuities and incompatible boundary conditions. As is well known, solutions are generally not smooth in these cases, unless the data satisfy appropriate conditions. Also not covered are singular parametrizations as well as non-smooth weight functions. The latter arise when applying Rvachev's technique for domains with corners (at least if the simplest R-function system is used). In this case one is confronted with lack of regularity of solutions as well and should note that using high degree b-splines (isogeometric or weighted) will not pay off for non-smooth problems (corners, discontinuities, singularities, etc.) unless suitable enhancements are used (e.g., additional special basis functions or adaptive refinement).

We have considered a single parametrization Φ since this is the case which is most relevant for the weighted isogeometric method. With our handling of trim surfaces one can often avoid partitioning the domain. When several parametrizations are used (cf. Figure 6), the quasi-interpolation error has to be estimated separately for each deformed hyper-rectangle. Since, in general, parametrizations will merely join continuously, composition with Φ or Φ^{-1} is no longer globally bounded with respect to Sobolev norms. Hence, the arguments become much more elaborate, already without a weight function (see [2] for a comprehensive error analysis as well as a number of open questions mentioned in the introduction).

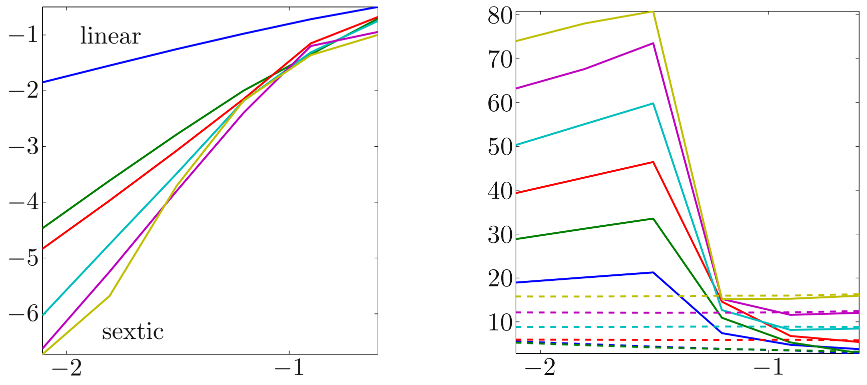


Fig. 9. Logarithmic plots of H^1-errors (left) and condition numbers (right) of weighted isogeometric Ritz-Galerkin approximations for the mixed Poisson problem with $f = 1$

We confirm the asserted convergence rates experimentally. The left diagram of Figure 9 shows the expected increase in accuracy with the degree. The logarithmic errors $\log \|u - u_h\|_{1,D}$ are plotted as functions of $\log h$; the labels show the decimal exponents. From the slopes we obtain estimates for the average rate, e.g., the rate approaches 5 for quintic approximations (degree 5). The computations were performed without extension, i.e., with the simple weighted isogeometric basis. We note that the instability has virtually no effect on the accuracy, despite huge condition numbers, as shown in the right diagram (solid lines). For example, the condition, estimated with the MATLAB `condest` command, exceeds 10^{70} for degree 6. This limited relevance of stability has been noticed in other examples. The phenomenon is probably due to the fact that preconditioning is inherent in many iterative solvers like the pcg-ssor routine used for this example. A theoretical explanation is given by B. Mößner and U. Reif [16], who showed that bivariate b-splines can be stabilized simply by scaling. Hence, at least in two variables, the extension procedure described at the end of Section 4 is, also from a theoretical point of view, not necessary.

The right diagram also shows the condition numbers for stabilized weighted extended isogeometric bases for degrees 1 to 6 (dashed lines at the bottom of the diagram). As expected the values are drastically smaller, reflecting the predicted order $O(h^{-2})$ for the condition of the Ritz-Galerkin system. In all cases, the condition numbers for the extended bases are $\leq 10^{20}$. Perhaps somewhat surprisingly, not only the errors but also the number of pcg-ssor iterations are roughly identical for the stable and unstable case in this example; another indication of inherent preconditioning.

8 Applications

The true measure for comparing finite element software are complicated three-dimensional problems. We give examples for the weighted and weighted isogeometric b-spline techniques. For the pure isogeometric method we refer to the literature (cf., e.g., [7] and the references cited therein) since we have at this point only the simple one-patch discretization implemented which cannot handle complex geometrical structures.

Figure 10 shows a domain D with many holes, which is defined implicitly by a randomly generated piecewise trilinear weight function

$$D: \ w_h = \sum_k w_k b_k > 0 \, .$$

Clearly, only the weighted method will handle the complicated boundary pattern efficiently. As a test case, we solve the Poisson problem with essential homogeneous boundary conditions and

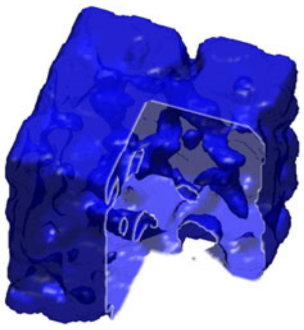
$$\mathcal{Q}(u) = \frac{1}{2} \int_D |\mathrm{grad} u|^2 - 2u \, .$$

Due to the irregular boundary, the regularity of the solution is poor. Hence, linear b-splines provide an adequate approximation:

$$u_h = w_h p_h, \quad p_h = \sum_k u_k b_k \, .$$

Using the same representation for the weight function w_h as well as for the spline p_h, leads to an appealing data structure. On each grid cell (cube with edge length h), the approximation u_h is determined by 2×2^3 values at the corners which coincide with the b-spline coefficients in this case. As a consequence, as is described in [13], a very efficient solution procedure is possible. For example, on a grid of $577^3 = 192,100,033$ unknowns, a dynamic vectorized multigrid solver on a NEC SX8 with 8 CPUs on one node reaches a relative residual less than $1E - 8$ in < 50 seconds of real time. Perhaps even more striking is that the time for assembling the discrete finite element system (usually the bottle neck in simulations) is comparable to the time required by the solver. This demonstrates the excellent performance of assembly algorithms for b-spline based methods.

It is interesting to note that the multigrid iteration has been programmed without stabilization via extension. While stability of the basis is (to date!) essential for the convergence theory, our solver does not seem to require it; a simple diagonal preconditioning suffices, at least for this particular application (cf. also the remarks at the end of Section 7 and [16]).



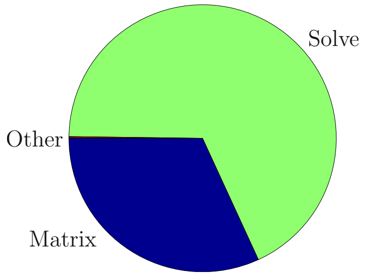


Fig. 10. Randomly generated domain for Poisson's problem and relative computing time of program components

In our second example, we consider the deformation of an elastic solid occupying a volume $D \subset \mathbb{R}^3$ under gravity. The displacement

$$(u_1(x), u_2(x), u_3(x)), \quad x \in D,$$

satisfies the Navier-Lamé system and minimizes the energy functional

$$\mathcal{Q}(u) = \frac{1}{2} \int_D \sigma(u) : \varepsilon(u) - fu,$$

where σ is the stress and ε the strain tensor.

We choose a geometric form which is neither ideally suited for weighted nor for isogeometric b-spline approximations. The curved bridge shown in Figure 12 has a relatively simple shape. Nevertheless both, the weighted and the isogeometric method, do not provide good discretizations as is illustrated in Figure 11. A straightforward description of the domain with a global weight function leads to unnecessary many boundary cells. On the other hand, while the principal shape is an absolutely elementary example of a deformed cuboid (a Bézier parametrization of coordinate degree $(1, 1, 2)$ suffices), the trim surfaces prevent the representation with a simple parametrization. Instead, a partition into deformed cubes is required (cf. [7], Figure 2.29 for a similar example), which does not reflect the simplicity of the geometry. As a consequence, one loses some of the computational efficiency; a vectorizable assembly routine is no longer applicable in a straightforward fashion.

The mixed method described in Section 6 suggests itself. We use a single polynomial parametrization Φ in Bézier form on a rectangle R and model the trim surfaces by a product of three elementary weight functions. In other words, we use finite elements of the form

$$e_\nu \left(\prod_{\alpha=1}^{3} w_\alpha(\xi) \right) b_k(\xi), \quad \nu = 1, 2, 3,\ k \sim R,$$

where the multiplication by the unit vectors e_ν takes the vector-valued form of the approximation u_h into account. As a consequence, we obtain fairly accurate

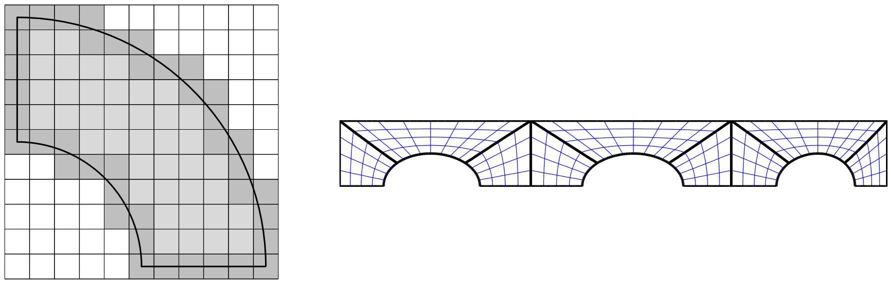


Fig. 11. Weighted (left, top view) and isogeometric (right, side view) discretization of a curved bridge

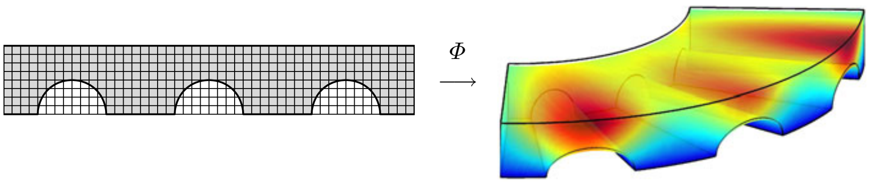


Fig. 12. Deformation of an elastic solid

numerical results with relatively few parameters. Figure 12 visualizes the magnified deformation for concrete (Young's modulus: $E = 50\,\mathrm{kN/mm}^2$, Poisson's ratio: 0.2). It was computed with $3 \cdot 50 \cdot 5 \cdot 9$ triquadratic finite elements ($3 \cdot 90$ of them are outside the trimmed domain). Only a section of the grid in the xz-plane is shown. Using a pcg solver, the system was solved to a relative accuracy of $< 1E - 10$ in less than 600 iterations.

The efficiency of b-spline algorithms, demonstrated by the above examples, is also reflected in the simplicity of finite element codes. The beauty of b-spline programming is particularly evident for multigrid techniques [11], where subdivision serves as canonical grid transfer.

9 Conclusion

Comparing Figures 5, 6, and 7, the various basis functions are qualitatively very similar. Perhaps this is not too surprising; after all, each of the slightly different concepts is based on b-splines. As a consequence, the finite elements share many advantages:

– free choice of smoothness and order of accuracy
– efficient recurrences for basic operations
– flexible geometry representation
– vectorized algorithms and multilevel techniques
– simple data structure with one node per grid point
– natural adaptive refinement via hierarchical bases

With many common favorable features, weighted and isogeometric methods both perform well for a broad range of applications. However, there are some pros and cons which we would like to point out below.

If a natural weight function is available, like for constructive solid geometry models or for many problems in linear elasticity, weighted approximations suggest themselves. Another ideal application are free boundary problems, where the unknown boundary can be implicitly represented by a spline serving as a weight function which changes with time.

On the negative side, the numerical construction of weight functions from boundary representations can be difficult, particularly in three dimensions. General purpose schemes have to rely on efficient algorithms for computing the distance function in a neighborhood of the domain boundary.

Isogeometric techniques are the method of choice if a CAD description of the domain D as solid model without trimming is available. This means that D is already partitioned into moderately deformed rectangles and cubes, respectively. If this is not the case, depending on the topological structure of D, the domain decomposition can be nontrivial. There is some similarity to the generation of coarse hexahedral meshes. Of course, we could also allow triangles and tetrahedra as additional parameter domains. But then, the isogeometric method loses some of its computational efficiency.

Numerical integration is crucial for both techniques. Isogeometric methods require only integration over grid cells, an ideal situation. However, the evaluation of integrands of the form (3) is time consuming. Hence, minimal node formulas are especially important. Weighted methods have simpler integrands. Yet, the integration over boundary cells is nontrivial. To preserve the high accuracy of the b-spline approximations, cells which are intersected by the domain boundary must be partitioned into smooth images of standard domains. Fortunately, topologically difficult intersection patterns are less frequent so that they do not have a significant impact on the overall computing time.

The mixed method introduced in this paper eliminates some of the difficulties mentioned above. Figure 7 serves as a typical example. Boundary integration is necessary only for grid cells with relatively simple intersection patterns. Moreover, despite the nontrivial shape of the domain, computations use a parametrization over a single rectangle.

As is apparent from the above remarks, there are numerous topics for future research. With a joint effort, continued progress will be made, and we believe that b-splines have the potential to become a standard in finite element analysis, too.

Acknowledgements. We thank Ulrich Reif and Joachim Wipper, the co-founders of the web-method, for a continuing excellent cooperation. Moreover, we gratefully acknowledge the support by the numerical analysis group in Stuttgart. In particular, we thank Christian Apprich for valuable suggestions and for maintaining our two-dimensional web-software. As usual, Mrs. Walter helped us with a careful reading of our manuscript.

References

1. Apaydin, G.: Finite Element Method with Web-splines for Electromagnetics. VDM Verlag, Saarbrücken (2009)
2. Bazilevs, Y., Beirão da Veiga, L., Cottrell, J.A., Hughes, T.J.R., Sangalli, G.: Isogeometric Analysis: Approximation, stability and error estimates for h-refined meshes. Math. Models Meth. Appl. Sci. 16, 1031–1090 (2006)
3. Bazilevs, Y., Calo, V.M., Cottrell, J.A., Evans, J., Lipton, S., Hughes, T.J.R., Scott, M.A., Sederberg, T.W.: Isogeometric Analysis using T-Splines. Comput. Methods Appl. Mech. Engrg. 199, 229–263 (2010)
4. Bazilevs, Y., Calo, V.M., Hughes, T.J.R., Zhang, Y.: Isogeometric fluid-structure interaction: Theory, algorithms and computations. Comput. Mech. 43, 3–37 (2008)
5. de Boor, C.: A Practical Guide to Splines. Springer, New York (1978)
6. Calderón, A.P.: Lebesgue spaces of differentiable functions and distributions. Proc. Symp. in Pure Math. 4, 33–49 (1961)
7. Cottrell, J.A., Hughes, T.J.R., Bazilevs, Y.: Isogeometric Analysis: Toward Integration of CAD and FEA. Wiley, Chichester (2009)
8. Gomez, H., Hughes, T.J.R., Nogueira, X., Calo, V.M.: Isogeometric analysis of the isothermal Navier-Stokes-Korteweg equations. Comput. Methods Appl. Mech. Engrg. 199, 1828–1840 (2010)
9. Hoffacker, A.: Gewichtete Isogeometrische Finite Elemente mit B-Spline-Ansatzfunktionen. Master Thesis, Stuttgart (2011)
10. Höllig, K.: Finite Element Methods with B-Splines. SIAM, Philadelphia (2003)
11. Höllig, K., Hörner, J.: Programming multigrid algorithms with b-splines (in preparation)
12. Höllig, K., Reif, U., Wipper, J.: Weighted extended B-spline approximation of Dirichlet problems. SIAM J. Numer. Anal. 39, 442–462 (2001)
13. Höllig, K., Hörner, J., Pfeil, M.: Parallel finite element methods with weighted linear b-splines. In: Nagel, W.E., Kröner, D., Resch, M. (eds.) High Performance Computing in Science and Engineering 2007, pp. 667–676. Springer, Heidelberg (2008)
14. Hughes, T.J.R., Cottrell, J.A., Bazilevs, Y.: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. Comput. Methods Appl. Mech. Engrg. 194, 4135–4195 (2005)
15. Kantorowitsch, L.W., Krylow, W.I.: Näherungsmethoden der Höheren Analysis. VEB Deutscher Verlag der Wissenschaften, Berlin (1956)
16. Mößner, B., Reif, U.: Stability of tensor product b-splines on domains. J. Approx. Theory 154, 1–19 (2008)
17. Rvachev, V.L., Sheiko, T.I.: R-functions in boundary value problems in mechanics. Appl. Mech. Rev. 48, 151–188 (1995)
18. Schumaker, L.L.: Spline Functions: Basic Theory. Wiley-Interscience, New York (1980)
19. Stein, E.M.: Singular Integrals and Differentiability Properties of Functions. Princeton University Press, Princeton (1970)
20. Strang, G., Fix, G.J.: An Analysis of the Finite Element Method. Prentice–Hall, Englewood Cliffs (1973)
21. Zienkiewicz, O.C., Taylor, R.I.: Finite Element Method, vol. I–III. Butterworth & Heinemann, London (2000)

$\sqrt{3}$-Based 1-Form Subdivision

Jinghao Huang and Peter Schröder

Applied & Computational Mathematics, Caltech
1200 E. California Blvd., MC305-16, Pasadena, CA 91125, USA
huangjh@caltech.edu

Abstract. In this paper we construct an edge based, or *1-form*, subdivision scheme consistent with $\sqrt{3}$ subdivision. It produces smooth differential 1-forms in the limit. These can be identified with tangent vector fields, or viewed as edge elements in the sense of finite elements. In this construction, primal (0-form) and dual (2-form) subdivision schemes for surfaces are related through the exterior derivative with an edge (1-form) based subdivision scheme, amounting to a generalization of the well known formulé de commutation.

Starting with the classic $\sqrt{3}$ subdivision scheme as a 0-form subdivision scheme, we derive conditions for appropriate 1- and 2-form subdivision schemes without fixing the dual (2-form) subdivision scheme a priori. The resulting degrees of freedom are resolved through spectrum considerations and a conservation condition analogous to the usual moment condition for primal subdivision schemes.

Keywords: Discrete exterior calculus, $\sqrt{3}$-subdivision, 1- and 2-form subdivision, tangent vector field generation.

1 Introduction

Subdivision algorithms for the construction of smooth free form surfaces are now well established [1,2,3] and have found applications well beyond geometric modeling [4,5]. Most of the schemes proposed so far are based on meshes with the data living at either vertices, *e.g.*, positions, or faces, *e.g.*, colors [6,7,8,9,10,11]. Such primal and dual schemes respectively come in many flavors distinguished by their topological split rules, *e.g.*, based on triangles or quadrilaterals, and the geometric smoothing rules, *e.g.*, piecewise linear or higher order. Less well studied are subdivision schemes based on data living on edges [12]. In this setting scalar coefficients associated with directed edges in the coarser mesh are linearly combined to give the new scalar coefficients on edges in the refined mesh. Using a suitable interpolation method for these coefficients one can construct a sequence of everywhere defined differential 1-forms which, under suitable conditions, converge to a limit. Hence such edge based subdivision schemes are a natural fit for the construction of differential 1-forms. These in turn may be identified with tangent vector fields [12]. They can also be seen as giving rise to *edge elements* in the sense of finite elements, which can be essential in the numerical resolution of certain partial differential equations [14,15]. In this way edge based subdivision schemes provide new construction methods for hierarchically refinable edge elements, for example.

$\sqrt{3}$-subdivision [10] was the first triangle-based subdivision scheme with a non-traditional topological refinement step. Instead of quadrisecting triangles, *i.e.*, inserting a new vertex for each edge, it inserts a new vertex for each triangle and then connects these with the old vertices. One of its distinguishing features is the slower growth rate of triangles from the coarser to the finer mesh (factor of 3 rather than 4). $\sqrt{3}$-subdivision as proposed by Kobbelt generates $\mathcal{C}^2$surfaces everywhere, but at the irrgular vertices (where it is only $\mathcal{C}^1$), while using only a small stencil. Notably, it allows for a smoother mesh gradation when adaptive refinement is used.

The first construction of a 1-form subdivision scheme was given for Loop subdivision [9] (a similar construction also applies to Catmull-Clark subdivision). There the stencil for the edge-based subdivision is derived by enforcing *commutative relations*

$$d^0 S_0 = S_1 d^0$$

These state that taking the (discrete) exterior derivative after subdivision should yield the same result as taking it before and then using the *associated* subdivision scheme. Fixing Loop as the 0-form (function) subdivision scheme and (generalized) half-box splines as the dual (face based) subdivision scheme uniquely fixes a smooth 1-form subdivision scheme on edges.

In this paper, we study the construction of a 1-form scheme for $\sqrt{3}$ refinement fixing only the primal subdivision scheme a priori to be the one of Kobbelt [10]. No *2-form* subdivision is specified in advance. In this case the commutative relations do not uniquely fix the scheme and we use spectrum and moment considerations to fix the scheme.

2 Mathematical Setup

We will be working with orientable 2-manifold triangle meshes of arbitrary topology. Such meshes are given as $\{V, E, T\}$, where $V = \{v_i\}$ represents all vertices (always oriented positively by convention), $E = \{e_{ij}\}$ all directed edges (e_{ij} goes from vertex i to vertex j) and $T = \{t_{ijk}\}$ all oriented facets (t_{ijk} has border orientation i, j, k). Furthermore we have point positions $P = \{p_i\}$, $p_i \in \mathbb{R}^3$ for each vertex which determine the embedding of the mesh through piecewise linear interpolation.

We also need the discrete analog of exterior calculus (DEC) [13] on a mesh. In DEC, smooth differential k-forms have their analog in k-cochains, *i.e.*, scalar coefficients associated with the k-simplices of the mesh (here $k = 0, 1, 2$ denotes the dimension of the underlying simplex). These coefficients represent the integral of the corresponding k-form over the underlying k-simplex (with integration over vertices interpreted as point evaluation):

$$\textit{0-form: } c_i = \omega^0(v_i), \quad \textit{1-form: } c_{ij} = \int_{e_{ij}} \omega^1, \quad \textit{2-form: } c_{ijk} = \int_{t_{ijk}} \omega^2. \tag{1}$$

Note that $c_{ij} = -c_{ji}$ and similarly $c_{ijk} = -c_{jik}$. That is, the orientation of the underlying simplex must be taken into account when manipulating the associated k-form coefficients. This will be important when applying the edge subdivision stencils, for example. Evaluating these integrals for a given smooth form ω over the appropriate simplices in the mesh then corresponds to projecting the given form into a subspace formed by as yet to be chosen bases.

With these in place the discrete analog of the smooth exterior derivative d is naturally defined as the co-boundary operator ensuring that Stokes' theorem is verified at the discrete level by construction

$$\int_{\Omega} d\omega = \int_{\partial\Omega} \omega. \tag{2}$$

At the discrete level of the mesh the integration domain on the left would be a 1- or 2-simplex, while on the right it is the corresponding boundary. This allows us to define the meaning of the discrete d in a purely topological manner (as it should be). For example, computing the discrete differential d^0 of a discrete 0-form (coefficients at vertices), one assigns to each edge e_{ij} the coefficients $c_{ij} = c_j - c_i$, *i.e.*, computes the signed sum of the coefficients at the edge boundary. Similarly, the discrete differential d^1 of a 1-form computes the discrete 2-form $c_{ijk} = c_{ij}+c_{jk}+c_{ki}$ at each triangle as the signed sum of coefficients over its boundary. Consequently, the linear operators d^0 and d^1 are realized through the signed incidence matrices of edges on vertices resp. triangles on edges (See Eq.(7)).

These discrete operations can be brought into correspondence with the smooth exterior derivative through suitably chosen reconstruction functions (forms). An example of this are the Whitney elements. For 0-forms, *i.e.*, for data living at vertices, these are just the standard linear hat functions ϕ_i associated to each vertex. These will be used for functions on the surface as well as the surface itself. Given the ϕ_i, linear interpolants for edge coefficients (1-forms) can be built as

$$\phi_{ij} = \phi_i \mathrm{d}\phi_j - \phi_j \mathrm{d}\phi_i. \tag{3}$$

Finally for 2-form data, *i.e.*, coefficients at faces, a suitable reconstruction are the piecewise constants

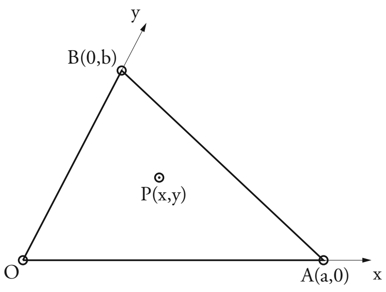
$$\phi_{ijk} = 2(\phi_i \mathrm{d}\phi_j \wedge \mathrm{d}\phi_k + \phi_j \mathrm{d}\phi_k \wedge \mathrm{d}\phi_i + \phi_k \mathrm{d}\phi_i \wedge \mathrm{d}\phi_j). \tag{4}$$

These particular bases ensure that exterior derivative and reconstruction commute: first reconstructing a 0-form and then applying the (smooth) exterior derivative yields the same result as first applying the discrete exterior derivative and then reconstructing with the ϕ_{ij}, and similarly for the transition from 1-forms to 2-forms.

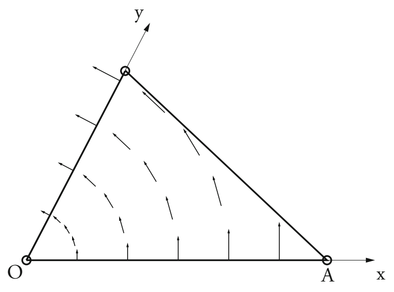
To visualize the 1-forms on the mesh we use a metric to associate 1-forms with (tangent) vector fields. Consider a single triangle embedded into an affine space (Fig. 1(a)), then

$$\begin{aligned} \mathrm{d}\phi_A &= \frac{1}{a}(1,0),\ \mathrm{d}\phi_B = \frac{1}{b}(0,1),\ \text{and } \phi_A = \frac{x}{a},\ \phi_B = \frac{y}{b} \\ \Rightarrow \phi_{AB} &= \frac{1}{ab}(-y,x) \perp (x,y), \end{aligned} \tag{5}$$

i.e., the vector is perpendicular with OP and has the magnitude $\frac{|OP|}{ab}$.



(a) Embed a single triangle into an affine space



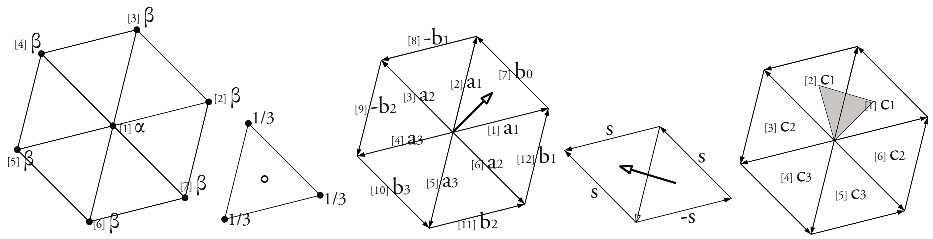
(b) Whitney 1-form ϕ_{AB}

Fig. 1. Visualization of a Whitney 1-form as a vector field

With these tools in place we can now proceed with the construction of the 1-form subdivision scheme. The key equations are the commutative relations (1): if we ensure that the discrete exterior derivative commutes with the subdivision operator *and* the corresponding subdivision scheme converges, then the limit 1-forms will form a basis for the underlying space of forms and discrete and smooth exterior derivatives will commute with reconstruction. This is entirely analogous to the formulé de commutation well known from the univariate setting and generalizes it to the 2-manifold setting in the context of subdivision. To be able to talk about a sequence of 1-forms converging, we introduced the piecewise linear Whitney interpolators ϕ_{ij} to map the coefficients on edges to a continuous 1-form. This is entirely analogous to the use of hat functions for interpolation of data at vertices to speak of a sequence of piecewise linear meshes converging to a smooth limit surface.

3 Regular Vertex

In this section we discuss the details of the subdivision stencil around regular vertices (valence $V = 6$). The subdivision scheme maps the coarse control mesh C_0 to the refined mesh C_1 and repeats this process iteratively. For the analysis we only need to consider a local invariant neighborhood, *i.e.*, a central vertex with its surrounding k-ring of regular vertices.



(a) *0-form*, odd/even vertices. (b) *1-form*, odd/even edges. (c) *2-form*.

Fig. 2. Stencils (odd/even) for 0-, 1-, and 2-form subdivision rules indicating the corresponding weights. The numbers in square brackets give the indices used in the subdivision matrices.

Using the indexing of Fig.(2) the associated subdivision matrices are:

$$S_0 = \left[\begin{array}{c|cccccc} \alpha & \beta & \beta & \beta & \beta & \beta & \beta \\ \hline 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 1/3 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 1/3 & 1/3 & 0 & 0 \\ 1/3 & 0 & 0 & 0 & 1/3 & 1/3 & 0 \\ 1/3 & 0 & 0 & 0 & 0 & 1/3 & 1/3 \\ 1/3 & 1/3 & 0 & 0 & 0 & 0 & 1/3 \end{array}\right]$$

$$S_1 = \left[\begin{array}{cccccc|cccccc} a_1 & a_1 & a_2 & a_3 & a_3 & a_2 & b_0 & -b_1 & -b_2 & b_3 & b_2 & b_1 \\ a_2 & a_1 & a_1 & a_2 & a_3 & a_3 & b_1 & b_0 & -b_1 & -b_2 & b_3 & b_2 \\ a_3 & a_2 & a_1 & a_1 & a_2 & a_3 & b_2 & b_1 & b_0 & -b_1 & -b_2 & b_3 \\ a_3 & a_3 & a_2 & a_1 & a_1 & a_2 & b_3 & b_2 & b_1 & b_0 & -b_1 & -b_2 \\ a_2 & a_3 & a_3 & a_2 & a_1 & a_1 & -b_2 & b_3 & b_2 & b_1 & b_0 & -b_1 \\ a_1 & a_2 & a_3 & a_3 & a_2 & a_1 & -b_1 & -b_2 & b_3 & b_2 & b_1 & b_0 \\ \hline -s & 0 & s & 0 & 0 & 0 & s & s & 0 & 0 & 0 & 0 \\ 0 & -s & 0 & s & 0 & 0 & 0 & s & s & 0 & 0 & 0 \\ 0 & 0 & -s & 0 & s & 0 & 0 & 0 & s & s & 0 & 0 \\ 0 & 0 & 0 & -s & 0 & s & 0 & 0 & 0 & s & s & 0 \\ s & 0 & 0 & 0 & -s & 0 & 0 & 0 & 0 & 0 & s & s \\ 0 & s & 0 & 0 & 0 & -s & s & 0 & 0 & 0 & 0 & s \end{array}\right]$$

$$S_2 = \begin{bmatrix} c_1 & c_1 & c_2 & c_3 & c_3 & c_2 \\ c_2 & c_1 & c_1 & c_2 & c_3 & c_3 \\ c_3 & c_2 & c_1 & c_1 & c_2 & c_3 \\ c_3 & c_3 & c_2 & c_1 & c_1 & c_2 \\ c_2 & c_3 & c_3 & c_2 & c_1 & c_1 \\ c_1 & c_2 & c_3 & c_3 & c_2 & c_1 \end{bmatrix} \qquad (6)$$

with $\alpha = 2/3$ and $\beta = 1/18$ for the regular case [10].

Using the same indexing convention the discrete exterior derivative operators have the following matrix representations:

$$d^0 = \left[\begin{array}{c|cccccc} -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 \end{array}\right], \quad d^1 = \left[\begin{array}{cccccc|cccccc} 1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right], \tag{7}$$

which are just the standard signed incidence matrices of the graph of the mesh.

Writing out the commutative relations between *0-* and *1-form* subdivision, $d^0 S_0 = S_1 d^0$, one finds 7 conditions:

$$\begin{cases} a_1 + a_2 + a_3 = 1/6 \\ a_1 + b_1 = 5/18 \\ b_0 = 0 \\ a_2 - b_1 + b_2 = -1/18 \\ a_3 - b_2 = -1/18 \\ b_3 = 0 \\ s = 1/6 \end{cases}, \tag{8}$$

only six of which are independent, leaving two free variables. We represent all variables in terms of the rim edge coefficients b_1 and b_2:

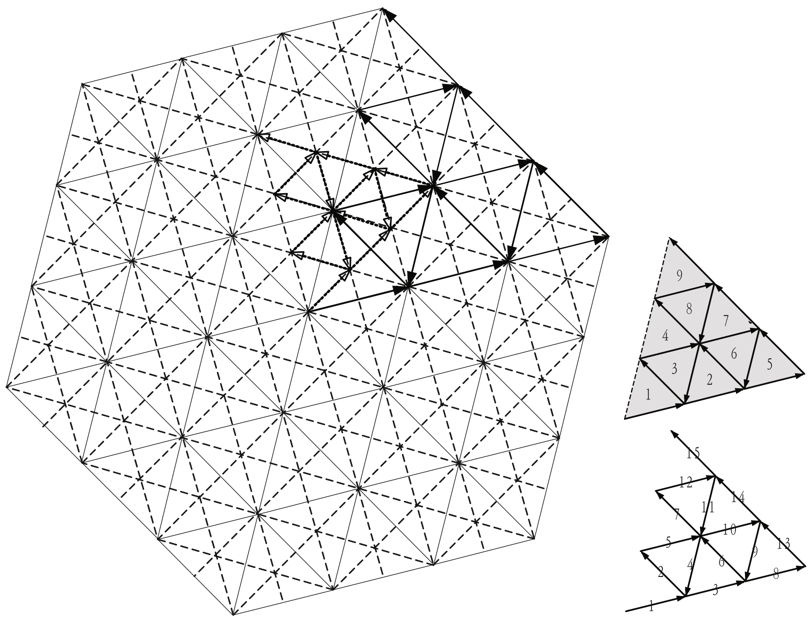
$$a_1 = \frac{5}{18} - b_1, \; a_2 = -\frac{1}{18} + b_1 - b_2, \; a_3 = -\frac{1}{18} + b_2. \tag{9}$$

Using the commutative relations between *1-* and *2-form* subdivision, $d^1 S_1 = S_2 d^1$, the parameters for the 2-form stencil may be written as functions of the 1-form stencil parameters

$$c_1 = -b_1 + s, \; c_2 = -b_2 + b_1, \; c_3 = b_2 = b_2. \tag{10}$$

In particular they do not add any constraints to b_1 or b_2.

To determine b_1 and b_2 we switch to the Fourier domain, using the generating function of the subdivision matrix. Fig.(3) shows the three ring invariant neighborhood around a regular vertex in both the coarse (C_0) and refined (C_1) meshes.



(a) *1-form*-subdivision (b) Indexing rule

Fig. 3. 3 rings of coarse mesh and refined mesh (before and after one step of $\sqrt{3}$-subdivision)

The corresponding subdivision matrix $S_1^{\mathbb{R}} \in \mathbb{R}^{90\times 90}$ can be simplified by taking advantage of the circulant symmetry of the k-ring neighborhood through a circulant Fourier transform giving a block diagonal matrix

$$\mathrm{diag}\{S_1^{\mathcal{F}}(z^0), S_1^{\mathcal{F}}(z^1), S_1^{\mathcal{F}}(z^2), \cdots, S_1^{\mathcal{F}}(z^{V-1})\},$$

where $z = \mathrm{e}^{2\pi\mathrm{i}/V}$ and

$$S_1^{\mathcal{F}}(z) = \begin{bmatrix} D_1 & 0 & 0 \\ * & D_2 & 0 \\ * & * & D_3 \end{bmatrix}$$

with

$$D_1 = \begin{bmatrix} a_1 + a_1 z + a_2 z^2 + a_3 z^3 + \frac{a_3}{z^2} + \frac{a_2}{z} & -b_1 z - b_2 z^2 + \frac{b_2}{z^2} + \frac{b_1}{z} \\ -\frac{1}{6} + \frac{z^2}{6} & \frac{1}{6} + \frac{z}{6} \end{bmatrix}$$

$$D_2 = \begin{bmatrix} -\frac{1}{6} & \frac{1}{6} & 0 & 0 & 0 \\ a_3 z & b_1 - a_3 z & a_2 - b_2 z & 0 & -b_2 \\ -a_3 z & -b_2 + a_2 z & -a_3 + b_1 z & -b_2 z & 0 \\ -a_2 z & a_3 z & -a_1 & b_2 z & b_1 \\ a_2 z & -a_1 z & a_3 & b_1 z & b_2 \end{bmatrix}$$

$$D_3 = \begin{bmatrix} 0 & -b_2 & -a_3 & a_3 & b_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_3 & -a_2 & -b_1 & 0 & b_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -b_1 z & -a_2 z & a_3 z & 0 & 0 & b_2 z & 0 \end{bmatrix} \tag{11}$$

Because the spectrum of $S_1^{\mathcal{F}}$ is formed by the eigenvalues of D_1, D_2 and D_3, the off-diagonal blocks are not important in the analysis which follows.

Because $\sqrt{3}$-subdivision performs a rotation of $\sqrt{z} = e^{\pi i/V}$ when going from the coarser to the finer mesh we expect to find a global phase factor of $\sqrt{z}$ in all eigen values [11], *i.e.*, they should be of the form $e^{j\pi i/V}\lambda$ for $\lambda \in \mathbb{R}$. This is certainly true for the D_1 and D_3 blocks. For the D_2 block this requirement implies further constraints on b_1 and b_2, leaving 3 possibilities:

Condition	a_1	a_2	a_3	b_1	b_2
Case 1: $b_2 = 0, b_1 = a_3$	1/3	−1/9	−1/18	−1/18	0
Case 2: $b_2 = 0, b_1 = -a_3$	2/9	0	−1/18	1/18	0
Case 3: $b_2 = -a_2, a_3 = 0$	2/9	−1/18	0	1/18	−1/18

To determine the choice of coefficients we use the generating function of the *2-form* subdivision (see Fig.(3) for the indexing rule for the facets in the inner 3 rings):

$$S_2^{\mathcal{F}}(z) = \left[\begin{array}{c|ccc|ccccc} c_3 z^3 + c_2 z^2 + c_1 z + c_1 + \frac{c_2}{z} + \frac{c_3}{z^2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline c_1 + c_2 z & c_3 z & c_1 + c_3 z & c_2 & 0 & 0 & 0 & 0 & 0 \\ z c_1 + c_1 & c_3 z & z c_2 + c_2 & c_3 & 0 & 0 & 0 & 0 & 0 \\ c_2 + c_1 z & c_2 z & c_3 + c_1 z & c_3 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & c_2 & c_1 & c_1 & 0 & c_3 & c_3 & c_2 & 0 \\ c_2 + c_3 z & c_2 z & c_1 + c_3 z & c_1 & 0 & 0 & 0 & 0 & 0 \\ z c_3 + c_3 & c_1 z & z c_2 + c_2 & c_1 & 0 & 0 & 0 & 0 & 0 \\ c_3 + c_2 z & c_1 z & c_3 + c_1 z & c_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & c_1 z & c_1 z & c_2 z & 0 & c_2 z & c_3 z & c_3 z & 0 \end{array}\right] \tag{12}$$

Each of the three choices for the coefficients gives a different spectrum for $S_2^{\mathcal{F}}$, only one of which has the correct leading terms (when ordered by modulus):

$$|\lambda_0| > \underbrace{|\lambda_1| = |\lambda_2|}_{\lambda_0/\sqrt{3}} > \cdots \tag{13}$$

This condition is satisfied only for case 2:

$$\begin{aligned} & c_1 = \frac{1}{9}, c_2 = \frac{1}{18}, c_3 = 0 \\ \Rightarrow \sigma(S_2^{\mathcal{F}}) &= \left\{ \frac{z^2}{18} + \frac{z}{9} + \frac{1}{18z} + \frac{1}{9}, \frac{\sqrt{z}}{18}, -\frac{\sqrt{z}}{18}, \frac{z}{18} + \frac{1}{18}, 0, 0, 0, 0, 0 \right\} \\ &= \{\frac{1}{3}, \frac{\sqrt{3}}{9} e^{\pm\frac{\pi i}{6}}, \cdots\} \end{aligned} \tag{14}$$

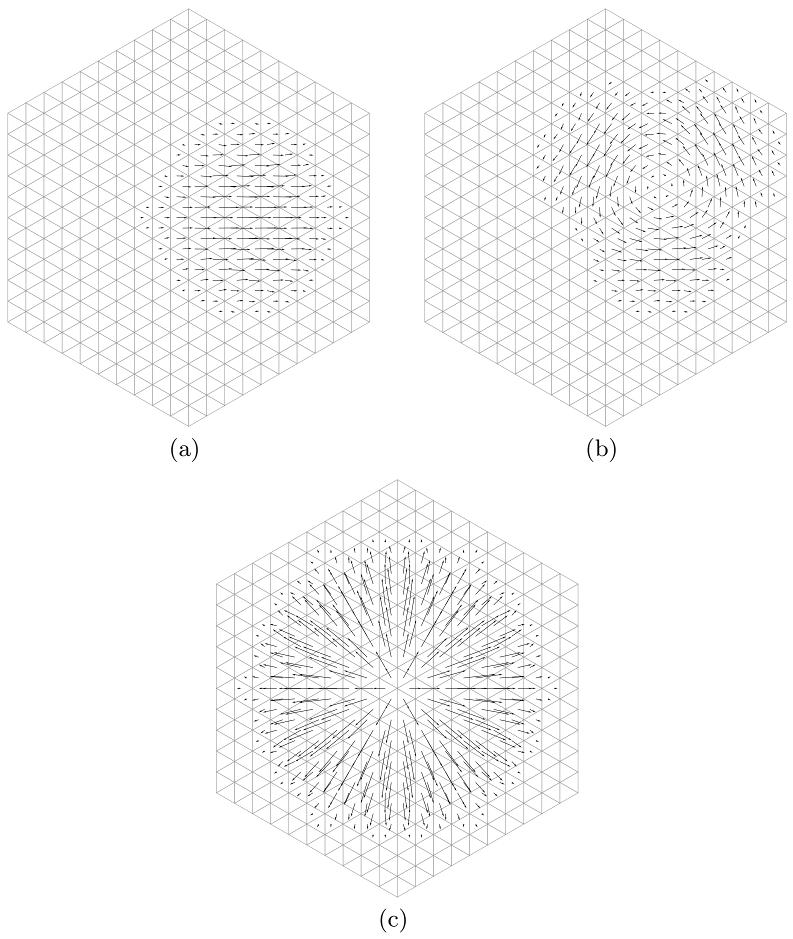


Fig. 4. Visualization of 1-form bases. After some number of subdivision levels, the 1-form over each triangle is visualized through its vector proxy at the barycenter of the triangle. (a) A single edge basis form (one edge coefficient is set to be 1); (b) three boundary edges of a coarse triangle with unit coefficient leading to a vortex; (c) source due to setting coefficients of all edges (at the coarsest level) incident to a single vertex to be 1.

Fig.(4) shows the subdivision process of a 1-form over a plane using the subdivision stencil above.

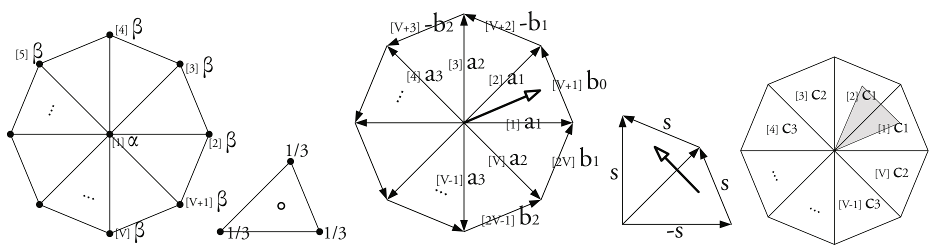
In summary, we used three steps to determine the subdivision stencil in the regular case: (1) the commutative relations between *0-* and *1-form* to reduce the system and leaving only two free parameters; (2) the *1-form* generating function $S_1^{\mathcal{F}}(z)$, with $z = \mathrm{e}^{2\pi \mathrm{i}/V}$, although complex-valued, should be representable as $\sqrt{z}$ times a real-valued matrix. With this condition there are only three possible choices for the stencil weights; (3) enforcing that the *2-form* generating function have a certain spectrum ([13]). This finally leads to a unique solution for the stencil weights.

4 Irregular Vertices

We now turn to determining the *1-form* and *2-form* subdivision stencils around irregular vertices. In this section, we begin with the general form of the problem before solving specific instances.

4.1 Basic Setup

Fig.(5) shows the variable settings of *0-*, *1-* and *2-form* subdivision stencils for arbitrary valences. The corresponding subdivision matrices follow the same pattern as those of the regular case (Eq.(2)).



(a) *0-form*, odd/even vertices. (b) *1-form*, odd/even edges. (c) *2-form*.

Fig. 5. The variable settings for all of the *0-, 1-* and *2-form* stencil (odd/even rules. extraordinary vertex case). The numbers in the square brackets provide the indexing rule we use to write down the subdivision matrices.

Proceeding as before with a circulant Fourier transform followed by enforcing the commutative relations we get (for $v = \lceil \frac{V}{2} \rceil$):

$$\left\{ \begin{array}{rcl} b_0 &=& 0 \\ b_v &=& 0 \text{ (V is even)} \\ a_1 &=& \frac{1}{3} - b_1 - \beta \\ a_2 &=& b_1 - b_2 - \beta \\ a_3 &=& b_2 - b_3 - \beta \\ & \cdots & \\ a_{v-1} &=& b_{v-2} - b_{v-1} - \beta \\ a_v &=& b_{v-1} - \beta \text{ when V is even, } a_v \;\; = \;\; 2b_{v-1} - \beta \text{ when V is odd} \\ && (\textit{For simplicity, we say } a_v = b_{v-1} - b_v - \beta, \\ && \;\; \textit{here } b_v = 0 \textit{ when } V \textit{ is even, } b_v = -b_{v-1} \textit{ when } V \textit{ is odd }) \\ s &=& 1/6 \end{array} \right.$$

respectively,

$$\left\{\begin{array}{rcl} c_1 & = & \frac{1}{6} - b_1 \\ c_2 & = & b_1 - b_2 \\ c_3 & = & b_2 - b_3 \\ & \cdots & \\ c_{v-1} & = & b_{v-2} - b_{v-1} \\ c_v & = & b_{v-1} \text{ (V is even)}, \quad c_v \;=\; 2b_{v-1} \text{ (V is odd)} \\ & & (\textit{For simplicity, we say } c_v = b_{v-1} - b_v, \\ & & \;\textit{here } b_v = 0 \textit{ when } V \textit{ is even, } b_v = -b_{v-1} \textit{ when } V \textit{ is odd} \;) \end{array}\right.$$

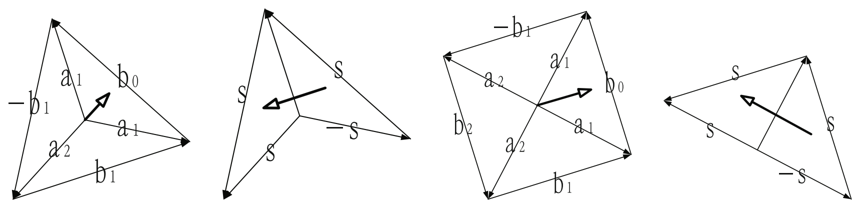
With $b_0 = 0$, $s = \frac{1}{6}$ and $b_v = 0$ (V even), we have $3v - 1$ stencil parameters:

$$a_1, a_2, \ldots, a_v; b_1, b_2, \ldots, b_{v-1}; c_1, c_2, \ldots, c_v,$$

of which the commutative relations each fix v parameters leaving us with $v - 1$ remaining free parameters.

4.2 Valence 3 and 4

For the case $V = 3$ and 4 all stencil variables can be solved for directly from the commutative relations and the spectrum requirement for the *2-form* subdivision (Eq. (13)).



(a) $V = 3$, even (b) $V = 3$, odd (c) $V = 4$, even (d) $V = 4$, odd

Fig. 6. The setting of variables

Based on the commutative relations the largest eigen value is always $\frac{1}{3}$ and comes from the upper-left (1×1)-block. The magnitude of the largest eigen value from any other block is always $\leq \frac{1}{9}$. Hence the subdominant eigenvalue can be choosen from the range $\left[\frac{1}{9}, \frac{1}{3}\right]$. For simplicity we set the subdominant eigen value to $\frac{\sqrt{3}}{9}$ as in the regular case (and use this choice always from now on).

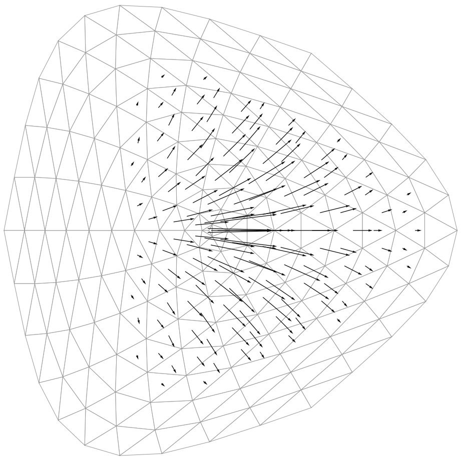
Fig.(6) shows the variable setting for $V = 3$ and 4. For the *0-form* stencil [10] $\alpha = 4/9$, $\beta = 5/27$, for $V = 3$ and we have

$$a_1 = \frac{5 + 2\sqrt{3}}{54}, a_2 = -\frac{2 + 2\sqrt{3}}{27}, b_1 = \frac{3 - 2\sqrt{3}}{54}, c_1 = \frac{3 + \sqrt{3}}{27}, c_2 = \frac{3 - 2\sqrt{3}}{27} \tag{15}$$

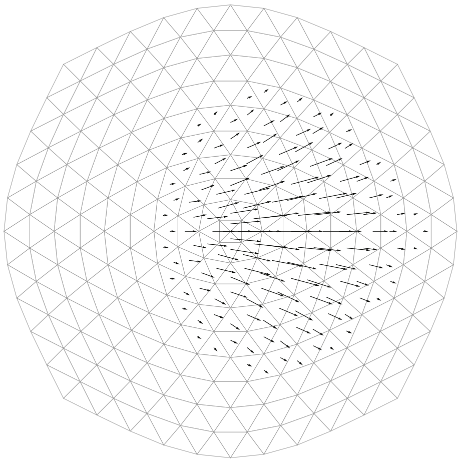
For $V = 4$, $\alpha = 5/9$, $\beta = 1/9$ and we get

$$a_1 = \frac{5 + \sqrt{6}}{36}, a_2 = -\frac{1 + \sqrt{6}}{36}, b_1 = \frac{3 - \sqrt{6}}{36}, c_1 = \frac{3 + \sqrt{6}}{36}, c_2 = \frac{3 - \sqrt{6}}{36} \tag{16}$$

The subdivision results for $V = 3, 4$ are visualized in Fig.(7).



(a) $V = 3$.



(b) $V = 4$.

Fig. 7. Basis 1-forms (visualized via their vector proxies) on an edge incident to a vertex of valance 3 (top) resp. 4 (bottom)

4.3 Irregular Vertex: V=5

For vertices with $V > 4$ ($V \neq 6$) the stencil is not completely determined by the techniques employed so far. Here we start with the case $V = 5$ and then generalize to $V > 6$.

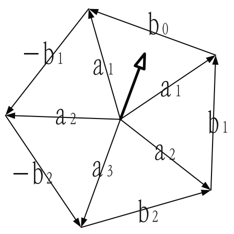


Fig. 8. $V = 5$. Stencil for the even edge in the case $V = 5$.

Fig.([8]) shows the variable setting for $V = 5$. Based on the commutative relation, we have

$$\begin{cases} a_1 = \frac{1}{3} - \beta - b_1 \\ a_2 = b_1 - b_2 - \beta \\ a_3 = 2b_2 - \beta \end{cases} \text{and} \begin{cases} c_1 = \frac{1}{6} - b_1 \\ c_2 = b_1 - b_2 \\ c_3 = 2b_2 \end{cases} \tag{17}$$

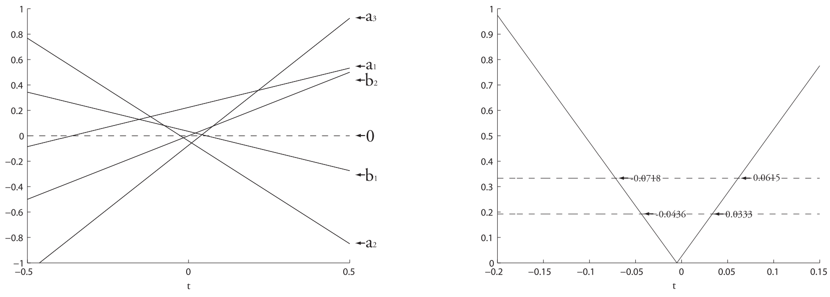
For the *0-form* stencil [10], when $V = 5$, $\alpha = \frac{9+\sqrt{5}}{18}, \beta = \frac{9-\sqrt{5}}{90}$. Further, similar to the low valence cases, we can choose the subdominant eigenvalue of the *2-form* subdivision generating function to be the same as in the regular case, *i.e.*, $\frac{\sqrt{3}}{9}$. Just as in the low valence case, this subdominant eigenvalue comes from the upper-left (1×1)-block of $S_2^{\mathcal{F}}$, *i.e.*, $g(z) \triangleq c_3z^3+c_2z^2+c_1z+c_1+c_2z^{-1}$. $g(1) = \frac{1}{3}$ is the largest eigen value. Letting $z_0 = \mathrm{e}^{2\pi\mathrm{i}/5}$, the subdominant eigenvalue is associated with the frequencies $z = z_0^{\pm1}$, *i.e.*, $|\lambda_1| = |g(\mathrm{e}^{2\pi\mathrm{i}/5})| = \frac{\sqrt{3}}{9}$. Using b_2 as the parameter t all stencil coefficients are functions of t

$$\sqrt{5}b_1 + \frac{5-\sqrt{5}}{2}b_2 = \frac{3-4\sqrt{3}+3\sqrt{5}}{36} \tag{18}$$

$$\Rightarrow \begin{cases} a_1 = -\frac{1-\sqrt{5}}{2}t + \frac{27-\sqrt{5}+4\sqrt{15}}{180}, \ a_2 = -\frac{1+\sqrt{5}}{2}t - \frac{3-5\sqrt{5}+4\sqrt{15}}{180}, \ a_3 = 2t - \frac{9-\sqrt{5}}{90} \\ b_1 = \frac{1-\sqrt{5}}{2}t + \frac{15+3\sqrt{5}-4\sqrt{15}}{180}, \ b_2 = t \\ c_1 = -\frac{1-\sqrt{5}}{2}t + \frac{15-3\sqrt{5}+4\sqrt{15}}{180}, \ c_2 = -\frac{1+\sqrt{5}}{2}t + \frac{15+3\sqrt{5}-4\sqrt{15}}{180}, \ c_3 = 2t \end{cases}$$

We first try to estimate the value of t. Fig.([9(a)]) shows the relation between the stencil parameters of the *1-form* subdivision stencil and the unknown parameter t. From Fig.([8]) and the relative alignment between edges we deduce that a_1 should have a large and positive value, a_3 to be negative and b_2 to be small. Based on Fig.([9(a)]), t should be in the interval $[-0.2, 0.05]$.

We now consider again the eigen values of the *2-form* subdivision. After circulant Fourier transform (as before) we find the generating function for the *2-form* subdivision matrix



(a) The relation between stencil parameters and t.

(b) The only eigenvalue affected by t.

Fig. 9. Estimation of the parameter

$$S_2^{\mathcal{F}} = \left[\begin{array}{c|ccc|ccccc} c_3z^3+c_2z^2+c_1z+c_1+\frac{c_2}{z} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \frac{z}{18}+\frac{1}{9} & 0 & \frac{1}{9} & \frac{1}{18} & 0 & 0 & 0 & 0 & 0 \\ \frac{z}{9}+\frac{1}{9} & 0 & \frac{z}{18}+\frac{1}{18} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{z}{9}+\frac{1}{18} & \frac{z}{18} & \frac{z}{9} & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & \frac{1}{18} & \frac{1}{9} & \frac{1}{9} & 0 & 0 & 0 & \frac{1}{18} & 0 \\ \frac{1}{18} & \frac{z}{18} & \frac{1}{9} & \frac{1}{9} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{z}{9} & \frac{z}{18}+\frac{1}{18} & \frac{1}{9} & 0 & 0 & 0 & 0 & 0 \\ \frac{z}{18} & \frac{z}{9} & \frac{z}{9} & \frac{1}{18} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{z}{9} & \frac{z}{9} & \frac{z}{18} & 0 & \frac{z}{18} & 0 & 0 & 0 \end{array}\right]. \tag{19}$$

Of the 45 eigen values t only controls those generated by the upper 1x1 block in the $z = z_o^{\pm 2}$ frequency. The dependency between the affected eigenvalue (solid line) and t is shown in Fig.(9(b)). The dashed lines show the two largest of the remaining eigen values. Since the absolute value of the varying (by t) eigen value should be smaller than $\lambda_0 = 1/3$, $t \in (-0.0718, 0.0615)$. This is consistent with the analysis based on Fig.(9(a)).

Experimenting with a range of parameters for t within the constraints spelled out above yielded no apparently smooth eigen forms for that eigen value. This is not surprising given the smoothness of the 0-form scheme. Instead we chose t so that the eigen values controlled by it vanish, resulting in $t = \frac{3-2\sqrt{3}}{90}$ and

$$\textit{1-form}\left\{\begin{array}{l} a_1 = \frac{12+\sqrt{3}+\sqrt{5}+\sqrt{15}}{90} \sim 0.2205 \\ a_2 = \frac{-3+\sqrt{3}+\sqrt{5}-\sqrt{15}}{90} \sim -0.0323 \\ a_3 = \frac{-3-4\sqrt{3}+\sqrt{5}}{90} \sim -0.0855 \\ b_0 = 0,\ \ b_1 = \frac{9-\sqrt{3}-\sqrt{15}}{90} \sim 0.0377 \\ b_2 = \frac{3-2\sqrt{3}}{90} \sim -0.0052 \\ s = \frac{1}{6} \end{array}\right. \qquad \textit{2-form}\left\{\begin{array}{l} c_1 = \frac{6+\sqrt{3}+\sqrt{15}}{90} \sim 0.1289 \\ c_2 = \frac{6+\sqrt{3}-\sqrt{15}}{90} \sim 0.0429 \\ c_3 = \frac{3-2\sqrt{3}}{45} \sim -0.0103 \end{array}\right.$$

Remark: An interesting question is how the parameter t affects eigen 1-forms. The generating function for the *1-form* subdivision scheme is the same as in the regular case, *i.e.*, Eq.(11) except in the D_1 block. Among all 75 eigen values, only 2 (or 1 pair) of them can be affected by t. These affected eigen values, same as in the *2-form* case, are associated with the frequencies $z = z_0^{\pm 2}$ and come from the upper-left (2×2)-block, D_1.

The resulting *1-form* basis form is visualized in Fig.(10).

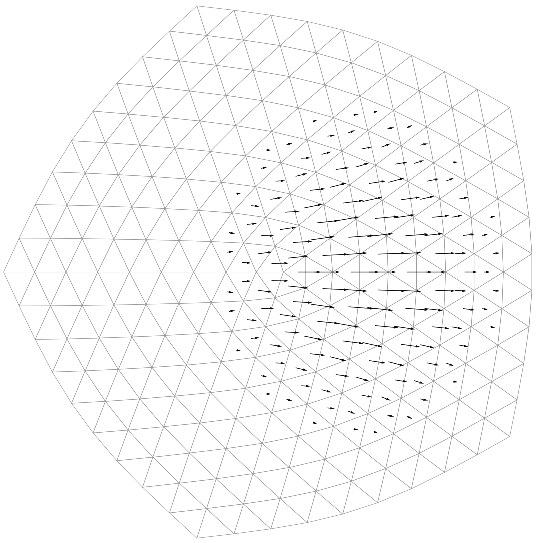


Fig. 10. Basis 1-form for an edge incident to an irregular vertex with $V = 5$

4.4 Irregular Vertex: Results

The case of $V > 6$ arbitrary valence proceeds very much like the $V = 5$ case. We consider again the resulting 2-form subdivision and the eigen values influenced by the remaining $v - 1$ parameters. All such eigen values arise from the upper left 1x1 block as before. Just as in the $V = 5$ case, after fixing the first 3 eigen values to be 1/3, $\sqrt{3}/9$ (up to the phase factor), all remaining eigen values that are controlled by the stencil parameters are set to zero.

Specifically, there are V eigen values influenced by the stencil parameters through $g(z) = c_1 + c_1 z + c_2 z^2 + c_3 z^3 + \ldots + \frac{c_3}{z^2} + \frac{c_2}{z}$. Call these $\bar{\lambda}_k$, where $k = 0, 1, \ldots, V-1$ indexes the block via $z_k = (\mathrm{e}^{2\pi \mathrm{i}/V})^k$.

(Frequency $\mathbf{z = z_0^0 = 1}$). Associated with the lowest frequency we get the first principal eigenvalue $\frac{1}{3}$. This is guaranteed by the commutative relations.

(Frequency $\mathbf{z = z_0^{\pm 1} = e^{2\pi i/V}}$). Associated with the second lowest frequencies, we get two second principal eigenvalues whose modulus, similarly to the regular and low-valence cases, are assumed to be $\frac{\sqrt{3}}{9}$:

$$|\lambda_1| = |g(\mathrm{e}^{2\pi \mathrm{i}/V})| = |c_1 + c_1 z + c_2 z^2 + c_3 z^3 + \ldots + \frac{c_3}{z^2} + \frac{c_2}{z}| = \frac{\sqrt{3}}{9}, \tag{20}$$

where $z = \mathrm{e}^{2\pi \mathrm{i}/V}$.

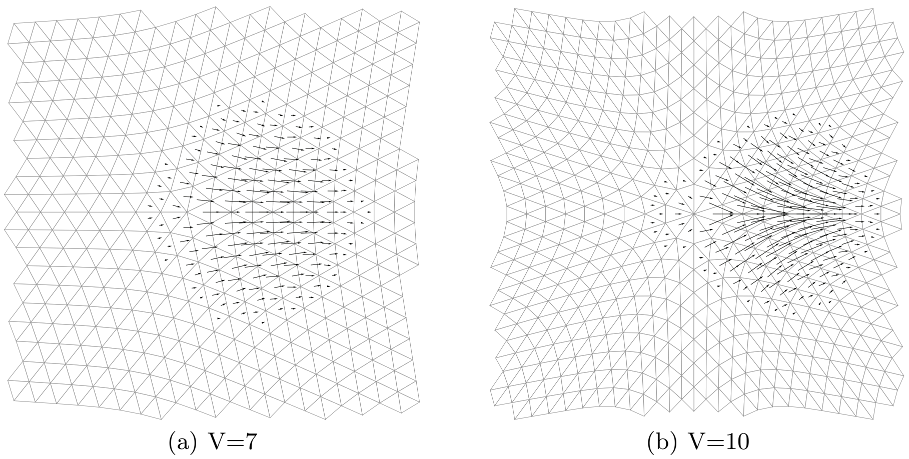
(Frequency $\mathbf{z = z_0^{\pm k} = e^{2k\pi i/V}, 2 \leq k \leq v - 1}$). We introduce $v-2$ independent linear equations by exactly the same method used in the valence-5-case. The idea is that in the real domain, all the affected eigenvalues associate with higher order of z_0 are zero. Specifically, for $k = 2, 3, \ldots, v-1$, $\bar{\lambda}^k = 0$, *i.e.*,

$$g(\mathrm{e}^{2k\pi \mathrm{i}/V}) = c_1 + c_1 z + c_2 z^2 + c_3 z^3 + \ldots + \frac{c_3}{z^2} + \frac{c_2}{z} = 0, \tag{21}$$

where $z = \mathrm{e}^{2k\pi \mathrm{i}/V},\ k = 2, 3, \ldots, v-1$.

There are just $v-1$ independent linear equations in Eqs.(21) and Eq.(20). Together with the commutative relations we can solve all the stencil parameters. We listed the results for $V = 7, 10, 16$ in Table.(1) below.

Finally, we use a 1-form over a plane which can contain vertex with arbitrary valence as examples. Fig.(11) shows the subdivision results for the plane containing a high-valence vertex in the center for $V = 7, 10$ and 16.


(a) V=7 (b) V=10

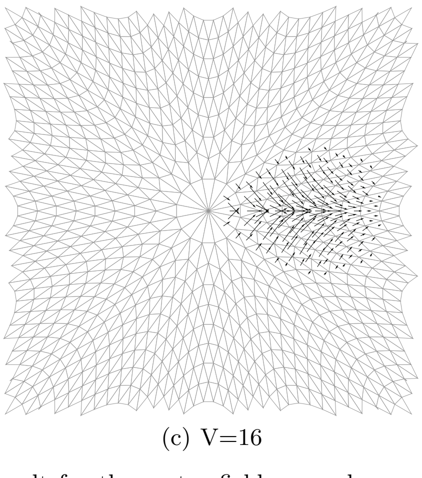

(c) V=16

Fig. 11. Subdivision result for the vector field on a plane containing a high-valence vertex

Table 1. Stencil coefficients: $V = 7, 10, 16$

V=7	a_1	a_2	a_3	a_4				
	0.220127	0.016156	-0.030363	-0.051065				
	b_1	b_2	b_3					
	0.069507	0.009653	-0.003683					
	c_1	c_2	c_3	c_4				
	0.097159	0.059855	0.013336	-0.007367				
V=10	a_1	a_2	a_3	a_4	a_5			
	0.210140	0.029491	0.006867	-0.015757	-0.029739			
	b_1	b_2	b_3	b_4	b_5			
	0.096727	0.040770	0.007437	-0.003273	0.000000			
	c_1	c_2	c_3	c_4	c_5			
	0.069940	0.055957	0.033333	0.010709	-0.003273			
V=16	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
	0.196148	0.025889	0.019252	0.010580	0.001194	-0.007478	-0.014115	-0.017707
	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8
	0.122239	0.081404	0.047206	0.021679	0.005539	-0.001929	-0.002761	0.000000
	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
	0.044427	0.040835	0.034198	0.025526	0.016140	0.007468	0.000831	-0.002761

5 Summary and Discussion

We have presented a *1-form* subdivision scheme based on $\sqrt{3}$ subdivision, which in our context is a *0-form* subdivision scheme. The construction of the subdivision stencil is based on the commutative relations and additional constraints on the spectrum. The *0-form* subdivision scheme used here is $\mathcal{C}^2$ in the regular setting but $\mathcal{C}^1$ around irregular vertices. The induced *1-form* scheme is also able to reproduce the linear vector fields in the regular setting. Around the extraordinary vertices, the *1-form* scheme can reproduce constant fields but not linear fields. This could be remedied with a larger *0-form* subdivision stencil. However, expanding the stencil eliminates the smallest-stencil advantage of the $\sqrt{3}$-subdivision *0-form* subdivision. The linear system will have more degree of freedom and can only be solved after introducing additional constraints on the stencil coefficients (*e.g.*, constraints on higher order terms in $\{\bar{\lambda}^k = g(z_0^k)\}$).

References

1. Zorin, D.: Stationary Subdivision and Multiresolution Surface Representations. PhD thesis, Caltech (1998)
2. Zorin, D., Schröder, P.: Subdivision for modeling and animation. Course Notes, ACM SIGGRAPH (2000)
3. Peters, J., Reif, U.: Subdivision Surfaces. Springer, Heidelberg (2008)
4. Khodakovsky, A., Schröder, P., Sweldens, W.: Progressive geometry compression. Comp. Graphics (ACM/SIGGRAPH Proc.), 271–278 (2000)
5. Grinspun, E., Krysl, P., Schröder, P.: Charms: A simple framework for adaptive simulation. ACM Trans. on Graph. 21(3), 281–290 (2002)

6. Reif, U.: A unified approach to subdivision algorithms near extraordinary vertices. Comput. Aided Geom. Design 12, 153–174 (1995)
7. Catmull, E., Clark, J.: Recursively generated b-spline surfaces on arbitrary topological meshes. Comp. Aid. Des. 10(6), 350–355 (1978)
8. Doo, D., Sabin, M.: Analysis of behaviour of recursive division surfaces near extraordinary points. Comp. Aid. Des. 10(6), 356–360 (1978)
9. Loop, C.: Smooth Subdivision Surfaces Based on Triangles. Master's thesis, University of Utah, Department of Mathematics (1987)
10. Kobbelt, L.: $\sqrt{3}$-Subdivision. Comp. Graphics (ACM/SIGGRAPH Proc.), 103–112 (2000)
11. Oswald, P., Schröder, P.: Composite primal/dual $\sqrt{3}$ subdivision schemes. Comput. Aided Geom. Design 20(5), 135–164 (2003)
12. Wang, K., Weiwei, T.Y., Desbrun, M., Schröder, P.: Edge subdivision schemes and the construction of smooth vector fields. ACM Transactions on Graphics 25(3), 1041–1048 (2006)
13. Grinspun, E., Schröder, P., Desbrun, M.: Discrete differential geometry: An Applied Introduction. Course Notes, ACM SIGGRAPH (2006)
14. Bossavit, A.: Computational Electromagnetism. Academic Press, Boston (1998)
15. Arnold, D.N., Falk, R.S., Winther, R.: Finite Element Exterior Calculus, Homological Techniques, and Applications. Acta Numerica 15 (2006)
16. Schlick, C.: An inexpensive brdf model for physically-based rendering. Comput. Graph. Forum 13(3), 233–246 (1994)
17. Turk, G.: Texture synthesis on surfaces. Comp. Graphics (ACM/SIGGRAPH Proc.), 347–354 (2001)

Curvature of Approximating Curve Subdivision Schemes

Kęstutis Karčiauskas[1] and Jörg Peters[2]

[1] Vilnius University
[2] University of Florida

Abstract. The promise of modeling by subdivision is to have simple rules that avoid cumbersome stitching-together of pieces. However, already in one variable, exactly reproducing a variety of basic shapes, such as conics and spirals, leads to non-stationary rules that are no longer as simple; and combining these pieces within the same curve by one set of rules is challenging. Moreover, basis functions, that allow reading off smoothness and computing curvature, are typically not available. Mimicking subdivision of splines with non-uniform knots allows us to combine the basic shapes. And to analyze non-uniform subdivision in general, the literature proposes interpolating the sequence of subdivision control points by circles. This defines a notion of discrete curvature for interpolatory subdivision. However, we show that this discrete curvature generically yields misleading information for non-interpolatory subdivision and typically does not converge, not even for non-uniform spline subdivision. Analyzing the causes yields three general approaches for solving or at least mitigating the problem: equalizing parameterizations, sampling subsequences and a new skip-interpolating subdivision approach.

Keywords: non-uniform subdivision, non-stationary subdivision, geometric continuity, curvature, splines, shape.

1 Introduction

A major selling point of subdivision algorithms has been their conceptual simplicity in smoothly connecting curve or surface regions by refinement with simple rules. A main task of curve modeling in product design is to reproduce segments of a variety of basic shapes, such as conics, spirals and clothoids *exactly*, and to transition smoothly between them. Since the standard uniform, polynomial subdivision algorithms cannot reproduce these basic shapes, a number of non-stationary curve subdivision algorithms have recently been devised to reproduce, in particular, circles and ellipses [15,20,3,6,18,4,7,2,19]. However, the introduction of parameter-dependent subdivision means that explicit basis functions for the control points are no longer easily available, removing a reliable technique to compute curvature. Already establishing smoothness or curvature continuity for non-stationary (or similar non-linear schemes [1,21,10,12,8]) is a challenge as general techniques, such as [5, p18] and [22], do not apply.

Furthermore, even when C^2 smoothness can be proven, this may be meaningless in practice without a reliable technique to compute curvature. Following e.g. Sabin et al.

[20,2], we may therefore attempt to compute a *discrete curvature*, as the reciprocal of a radius of the circle passing through the point and its two neighbours in the refined polygon. For interpolating curvature continuous schemes this measure must, by definition of a curvature continuous scheme, converge to the proper curvature. Also when refining the control polygon of uniform C^2 splines, the discrete curvature represents the curvature correctly. However, for the standard midpoint subdivision of non-uniform C^2 cubic splines, the measure diverges (cf. Fig. 4). In fact, divergence of the measure is typical as Lemma 1 shows.

This paper therefore discusses three techniques for estimating curvature from control points suitable for approximating subdivision: equalizing parameterization, sampling subsequences and skip-interpolation. To illustrate them, we work with non-uniform subdivision algorithms, a concept that is outlined in Section 2 and made concrete by a simple quadratic subdivision that can reproduce conics, in Section 6. An approach for reproducing more general basic shapes in one framework, is given in [14].

Concretely, for our focus on curvature from control points, we illustrate divergence of the discrete curvature for two non-uniform, polynomial spline subdivision schemes of degree 3, respectively degree 4. We also introduce the concept of skip-interpolation: every second control point is interpolated by the refined polygon. This allows measuring curvature as discrete curvature from the interpolating points while preserving the typically better shape of non-interpolating, approximating subdivision.

Overview. In Section 2, we explain the need for subdivision mimicking splines with non-uniform knots. In Section 3, we review non-uniform subdivision of degree 3 splines. In Section 4 we use this subdivision to analyze discrete curvature, prove divergence and to test strategies for obtaining predictive numbers from discrete curvature: equalizing parameters and subsampling. In Section 5, we derive a degree 4 skip-interpolating subdivision algorithm. In Section 6, we complete the exposition with an example of a non-uniform quadratic subdivision capable of reproducing various conics in one curve.

2 Non-uniform Subdivision

Non-uniform subdivision mimics the subdivision of splines with non-uniform knots. Fig. 1 and 2 illustrate the challenges that motivate non-uniform subdivision. Of the subdivision schemes listed in the introduction, Morin et al.'s scheme [15] is the only one that can reproduce more than one primitive in one curve. But, as Fig. 1 shows, even for this scheme, the underlying, inherently uniform spacing makes perturbations non-local. Fig. 2 (b) shows that the approach of [15] also unable to reproduce a circle on input of unevenly distributed samples. To prove the second claim, we note that for a regular polygon on the unit circle with opening angle α, the approach of [15] produces a circle of radius $\frac{\sin(\alpha)}{\alpha}$. If the designer's spacing of samples for perturbation is to be honored in the control polygon, the control points' distance to the circle center must be scaled so that they can partially reproduce the circle (*red* in Fig. 2(b)). In the transition, however, the reproduction is lost.

In order to reproduce different conics in one framework, splines use non-uniform knot sequences $\{t_i\}$. The rational cubic G^2 constructions in [14] combine curvature

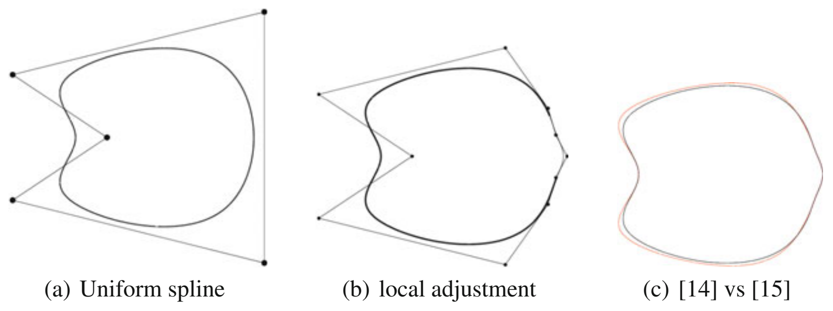


Fig. 1. In design, **local refinement and shape adjustment** naturally requires a switch from (a) uniform to (b) non-uniform spacing. The approach of [14] can preserve segments, shown as *thick segments*, of the original curve while (c) the superimposed *red* curve generated according to [15] deviates everywhere from the original curve, i.e. is less local.

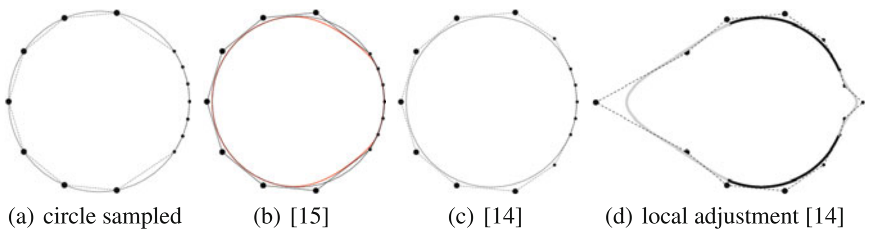


Fig. 2. Non-uniform spacing to support local adjustments. (a) Unequally-spaced designer samples on the circle in anticipation of local modification: small disks correspond to an opening angle of $\pi/16$, large ones to $3\pi/16$. (b) Circle in *black*, [15] in *red*. (c) Control net and exact circle generated by [14]. (d) Local modification using [14]: thick segments remain exactly on the circle.

continuity with exact reproduction of different basic shapes by simulating such non-uniformity of the knot sequence by a *non-uniform parameterization*. The basic approach can already be illustrated by a non-uniform quadratic subdivision mimicking rational G^1 splines. The details of such a scheme are given in Section 6. Here we outline the main idea. Let f_i and f_{i+1} be adjacent pieces of a C^1 spline with non-uniform knots, but with their domains re-parameterized to the unit interval $[0,1]$. Then f_i and f_{i+1} join with geometric continuity (see e.g. [13]):

$$f'_{i+1}(0) = \beta_i f'_i(1), \qquad \beta_i := \frac{\Delta_i}{\Delta_{i-1}}, \quad \Delta_i := t_{i+1} - t_i. \tag{1}$$

By refining the control structure of such splines, we arrive at non-uniform subdivision schemes [14] that are capable of combining primitives as shown in Fig. 1 and 2.

3 Non-uniform Subdivision of Cubic C^2 Splines

We now consider subdivision of non-uniform C^2 splines. Since we focus on measuring curvature from control polygons, we may restrict attention to the polynomial scheme,

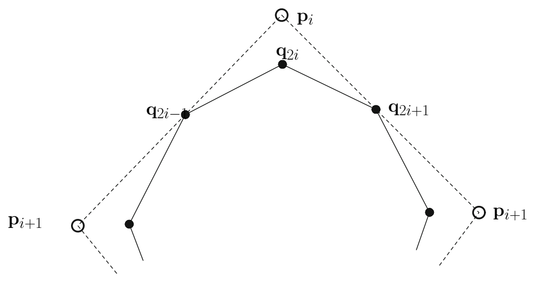


Fig. 3. $\mathbf{C^2}$ **cubic** control net refinement

rather than the more complex rational construction of [14] needed for reproducing basic shapes. Many draft shapes can be designed using B-splines with uniform knot sequence. However, subsequent *local* modifications lead to non-uniform splines as illustrated in Fig. 1. Fig. 3 illustrates the subdivision of such a non-uniform cubic B-spline curve. Suppose the cubic C^2 spline has the

$$\text{control polygon } \{\mathbf{p}_i\},\ \text{knot sequence } \{t_i\} \text{ and } \beta_i := \frac{\Delta_i}{\Delta_{i-1}},\ \Delta_i := t_{i+1} - t_i.$$

If we set $\bar{t}_{2i} := t_i$ and insert new knots at $\bar{t}_{2i+1} := (1-e_i)t_i + e_i t_{i+1}$, with ratio $0 < e_i < 1$ then the spline's refinement rules yield a new polygon $\{\mathbf{q}_i\}$, and constants,

$$\mathbf{q}_{2i+1} := (1-\eta_i)\mathbf{p}_i + \eta_i \mathbf{p}_{i+1}\ , \tag{2}$$

$$\mathbf{q}_{2i} := \mu_i \mathbf{q}_{2i-i} + (1-\mu_i-\nu_i)\mathbf{p}_i + \nu_i \mathbf{q}_{2i+i} \tag{3}$$

$$\eta_i := \frac{1+e_i\beta_i}{1+\beta_i+\beta_i\beta_{i+1}}\ ,\ \mu_i := \frac{\beta_i(1-e_i)}{1+\beta_i}\ ,\ \nu_i := \frac{e_{i-1}}{1+\beta_i}, \tag{4}$$

$$\beta_{2i}^{new} := \frac{e_i}{1-e_{i-1}}\beta_i, \quad \beta_{2i+1}^{new} := \frac{1-e_i}{e_i}. \tag{5}$$

Here, as in the previous section, we may interpret the terms β_i in (5) as constants of a linear reparameterization.

4 Discrete Curvature from Polygon Sequences

To be able to estimate curvature in the absence of explicit generating functions, we follow [20] in defining the *discrete curvature* to be the inverse radius of the circle interpolating three consecutive control points. The first experiment below demonstrates that discrete curvature does not converge when tracking the control polygon of a non-uniform cubic C^2 spline under midpoint subdivision. A similar failure of discrete curvature as estimator of curvature occurs for subdivision based on a quartic C^2 spline (Fig. 8). The particular setup of non-uniform cubic C^2 spline subdivision is helpful in that we can easily compute the true curvature of the limit curve of the limit curve.

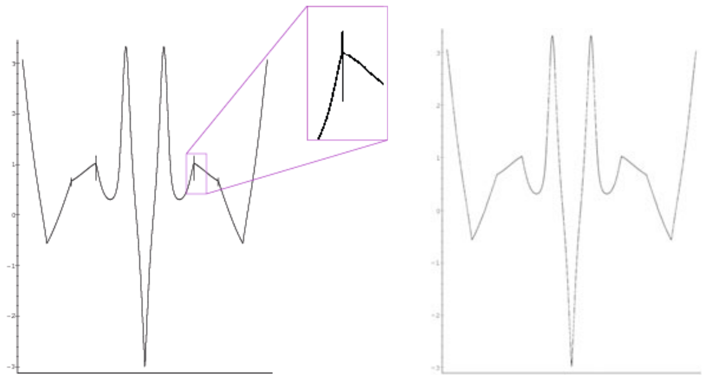


Fig. 4. Discrete curvature of Fig. 1,(b), plotted against the control point index i after $k = 10$ refinement steps of (*left*) subdivision for non-uniform subdivision (discontinuity enlarged), (*right*) when applying equalizing subdivision: no visible discontinuity.

Experiment 1 – divergence. We locally refine the non-uniform C^2 cubic B-spline of Fig. 1, *right*, according to Section 3, by inserting knots with ratio $e_i = 1/2$. The discrete curvature after 10 refinement steps is displayed in Fig. 4, *left*. The spikes hint at a discontinuity at the junction where $\Delta_i := t_{i+1} - t_i$ changes. Indeed, if we pick a knot $\bar{t}_i$,

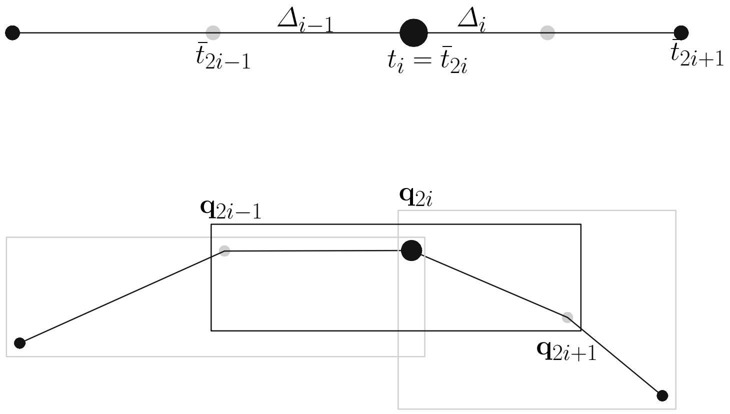


Fig. 5. Calculating the discrete curvature of a cubic C^2 spline from the control polygon

insert neighbors $\bar{t}_{2i-1}$ and $\bar{t}_{2i+1}$ using $e_{i-1} = e_i = 1/2$ so that the ratio $\beta_i = \Delta_i/\Delta_{i-1}$ remains unchanged under refinement, and set $\bar{t}_{2i} := \bar{t}_i$ as in Fig. 5, then there is no convergence. Let κ be the curvature of the spline at $\bar{t}_i$, denote by $\kappa_{2^k i-1}$ the discrete curvature to the left of $\bar{t}_i$ after k steps, by $\kappa_{2^k i+1}$ the discrete curvature to the right of $\bar{t}_i$ after k steps, and by $\kappa_{2^k i}$ the discrete curvature at $\bar{t}_i$. Since we have the underlying spline, we can compute the control polygon under midpoint subdivision and find

$$\lim_{k\to\infty} \kappa_{2^k i-1} = \frac{6}{\beta_i+5}\kappa\ , \ \lim_{k\to\infty} \kappa_{2^k i} = \kappa\ , \ \lim_{k\to\infty} \kappa_{2^k i+1} = \frac{6\beta_i}{5\beta_i+1}\kappa\ . \tag{6}$$

That is $\kappa_{2^k i}$ converges to κ but its left and right neighbors converge to the same value only if $\beta_i = 1$. In other words, only for uniform B-splines does the refined control polygon provide correct information on the curvature of the limit curve. Fig. 8 shows an even more extreme behavior for the control polygon of a spline of degree 4, defined by (12) in Section 5: the discrete curvature jumps everywhere. Generally, without care, we can therefore not infer the curvature directly from the control net of an approximating subdivision.

Experiment 2 – equalization. Since, according to the previous experiment, control polygons of uniform B-splines have useful discrete curvature expressions, we insert new knots $\bar{t}_{2i+1}$ in the spirit of [20]:

$$e_i := \frac{\sqrt{1+\beta_i}}{\sqrt{1+\beta_i}+\sqrt{\beta_i}\sqrt{1+\beta_{i+1}}}.$$

This moving ratio 'equalizes', i.e. we get, in the limit, a uniform knot sequence. Indeed, Fig. 4, *right*, shows that the spikes, hence discontinuities, in the discrete curvature disappear. The following lemma formally substantiates this observation.

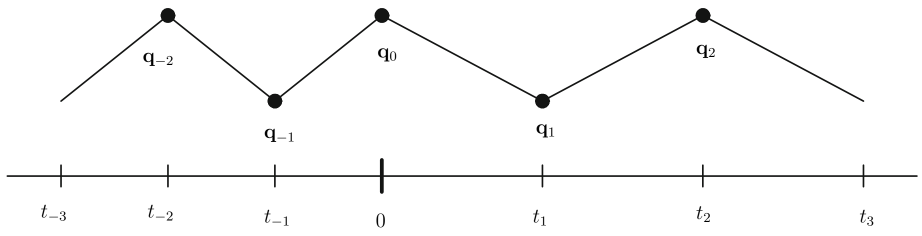


Fig. 6. Calculating the discrete curvature of a cubic G^2 spline

Lemma 1 (Continuity of the discrete curvature). *The discrete curvature of midpoint subdivision of a cubic G^2 B-spline with non-zero curvature is continuous if and only if its knot sequence is uniform; it is always continuous under equalizing subdivision.*

Proof. Let f be defined over $[-1, 0]$ and g over $[0, \beta]$. Consider subintervals near the origin 0 as illustrated in Fig. 6. For $\epsilon \to 0$ under subdivision,

$$f : [t_{-3}, t_{-2}], [t_{-2}, t_{-1}], [t_{-1}, 0], \tag{7}$$
$$t_{-3} := -\epsilon(h_{-1}+h_{-2}+h_{-3}), \quad t_{-2} := -\epsilon(h_{-1}+h_{-2}), \quad t_{-1} := -\epsilon h_{-1},$$
$$g : [0, t_1], [t_1, t_2], [t_2, t_3], \tag{8}$$
$$t_1 := \epsilon h_1, \quad t_2 := \epsilon(h_1+h_2), \quad t_3 := \epsilon(h_1+h_2+h_3).$$

For midpoint subdivision $h_{-3} = h_{-2} = h_{-1} = 1$, $h_1 = h_2 = h_3 = \beta$, while for equalizing subdivision, we have h_i depend on ϵ and $\lim_{\epsilon\to 0} h_i(\epsilon) = 1$. We compute the curvature κ of the spline at the origin 0; and we compute the discrete curvatures κ_i, $i = -1, 0, 1$, from triples $(\mathbf{q}_{i-1}, \mathbf{q}_i, \mathbf{q}_{i+1})$ of control points of the refined spline. Then with $h_i^\infty := \lim_{\epsilon\to 0} h_i(\epsilon)$ and $\rho_i := \lim_{\epsilon\to 0} \kappa_i/\kappa$,

$$\begin{aligned}\rho_{-1} =& \frac{3(h_{-2}^\infty + h_{-1}^\infty)}{h_{-3}^\infty + 2h_{-2}^\infty + 2h_{-1}^\infty + h_1^\infty}\ ,\ \rho_0 = \frac{3(h_{-1}^\infty + h_1^\infty)}{h_{-2}^\infty + 2h_{-1}^\infty + 2h_1^\infty + h_2^\infty}\ , \\ \rho_1 =& \frac{3(h_1^\infty + h_2^\infty)}{h_{-1}^\infty + 2h_1^\infty + 2h_2^\infty + h_3^\infty}.\end{aligned} \tag{9}$$

For midpoint subdivisions this yields as in (6)

$$\rho_{-1} = \frac{6}{5+\beta}\ , \rho_0 = 1\ ,\ \rho_1 = \frac{6\beta}{1+5\beta}, \tag{10}$$

while for equalizing subdivisions all $\rho_i = 1$ and the claim follows.

Note that the proof of Lemma 1 is based on the explicit knowledge of the underlying B-spline curve. A similar proof for a general non-stationary or non-linear subdivision scheme would be tricky.

Experiment 3 – subsequences. Since equalization leads to complicated subdivision rules, we try another approach. We insert new knots midway as in Experiment 1, but determine the discrete curvature from subsequences. If we choose every 2^3rd control point, the result for non-uniform C^2 subdivision looks gratifyingly like Fig. 4, *right*. However, proving convergence in a general setting is a subtle affair. By contrast, for the next approach it is straightforward.

5 Skip-Interpolating Subdivision

In order to obtain a subdivision scheme with easily measurable curvature, we generalize splines of degree 4, but such that every second control point stays fixed. Then the discrete curvature of the interpolating subsequence represents the limit curvature as it would for an interpolating subdivision algorithm.

First, we review subdivision of quartic C^2 splines. If all β_i equal 1 and the domain intervals of the Bézier quartics are split at their center then the refinement rules for obtaining new control points $[\bar{\mathbf{q}}, \tilde{\mathbf{q}}]$ from $[\bar{\mathbf{p}}, \tilde{\mathbf{p}}]$ are (cf. Fig. 7(a))

$$\begin{aligned}\bar{\mathbf{q}}_{2i} &:= \frac{3}{16}\tilde{\mathbf{p}}_{i-1} + \frac{5}{8}\bar{\mathbf{p}}_i + \frac{3}{16}\tilde{\mathbf{p}}_i, \quad \bar{\mathbf{q}}_{2i+1} := \frac{1}{8}\bar{\mathbf{p}}_i + \frac{3}{4}\tilde{\mathbf{p}}_i + \frac{1}{8}\bar{\mathbf{p}}_{i+1}\ , \\ \tilde{\mathbf{q}}_{2i} &:= \frac{1}{16}\tilde{\mathbf{p}}_{i-1} + \frac{3}{8}\bar{\mathbf{p}}_i + \frac{9}{16}\tilde{\mathbf{p}}_i, \quad \tilde{\mathbf{q}}_{2i+1} := \frac{9}{16}\tilde{\mathbf{p}}_i + \frac{3}{8}\bar{\mathbf{p}}_{i+1} + \frac{1}{16}\tilde{\mathbf{p}}_{i+1}\ .\end{aligned} \tag{11}$$

In Fig. 8, the control points $\bar{\mathbf{p}}_i$ (black disks) and $\tilde{\mathbf{p}}_i$ (gray disks) are equally distributed on the unit circle. The discrete curvature of the refined polygons in Fig. 8(c,d) shows

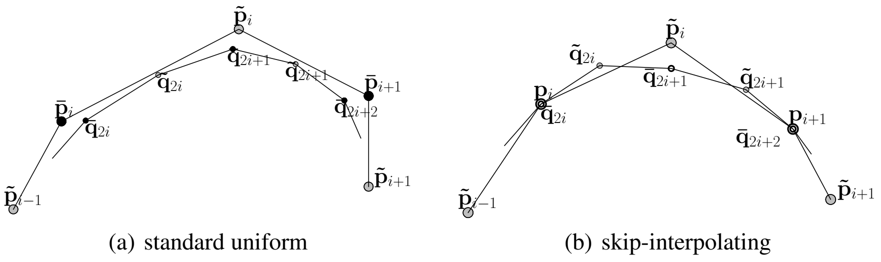


Fig. 7. Standard subdivision $[\bar{\mathbf{p}}, \tilde{\mathbf{p}}] \to [\bar{\mathbf{q}}, \tilde{\mathbf{q}}]$ and skip-interpolating subdivision $[\mathbf{p}, \tilde{\mathbf{p}}] \to [\mathbf{q}, \tilde{\mathbf{q}}]$

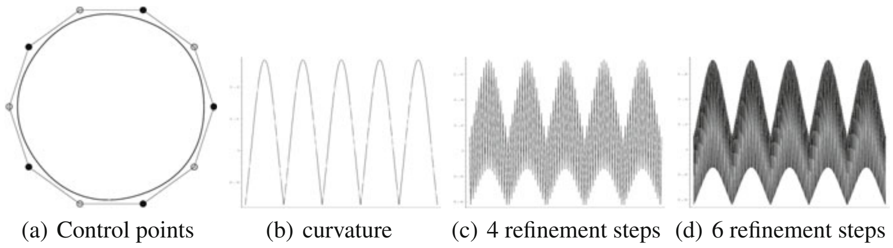


Fig. 8. (a) **Quartic C^2 spline**, (b) the spline's exact curvature plotted against the parameter on the absissa and (c,d) the discrete curvature at $i/2^k$ during the kth refinement. The discrete curvature jumps, oscillating densely about the true curvature.

that there is no sense in tracing their densely oscillating discrete curvature to estimate the curvature of the limit C^2 spline.

Next, we consider the conversion of the B-spline control points $\tilde{\mathbf{p}}_i$ of the polynomial spline to its Bernstein-Bézier coefficients $\mathbf{b}_{i,j}$ (see e.g. [11,17] for the definitions of the B-spline form and the Bernstein-Bézier (BB) form). With the constants β_i representing the ratios of the non-uniform lengths of adjacent knot intervals,

$$\mathbf{b}_{i-1,3} := \frac{\beta_i \tilde{\mathbf{p}}_{i-1} + \bar{\mathbf{p}}_i}{\beta_i + 1}, \qquad \mathbf{b}_{i1} := \frac{\beta_i \bar{\mathbf{p}}_i + \tilde{\mathbf{p}}_i}{\beta_i + 1}, \tag{12}$$
$$\mathbf{b}_{i,2} := \tilde{\mathbf{p}}_i, \qquad \mathbf{b}_{i-1,4} := \mathbf{b}_{i0} = \frac{\beta_i \mathbf{b}_{i-1,3} + \mathbf{b}_{i1}}{\beta_i + 1}.$$

In particular, for $\beta_i = 1$, we get the familiar formulas $\mathbf{b}_{i-1,3} := \frac{1}{2}\tilde{\mathbf{p}}_{i-1} + \frac{1}{2}\bar{\mathbf{p}}_i$, $\mathbf{b}_{i1} := \frac{1}{2}\bar{\mathbf{p}}_i + \frac{1}{2}\tilde{\mathbf{p}}_i$, $\mathbf{b}_{i-1,4} = \mathbf{b}_{i0} := \frac{1}{2}\mathbf{b}_{i-1,3} + \frac{1}{2}\mathbf{b}_{i1}$.

To arrive at skip-interpolation, we define $\mathbf{p}_i := \mathbf{b}_{i,0}$ and observe in (12) and Fig. 9 that the relation between the sequence of point triples $\tilde{\mathbf{p}}_{i-1}, \bar{\mathbf{p}}_i, \tilde{\mathbf{p}}_i$ and $\tilde{\mathbf{p}}_{i-1}, \mathbf{p}_i, \tilde{\mathbf{p}}_i$ is linear. Therefore, we can equally well express a subdivision with the structure of (11) in terms of points $\tilde{\mathbf{p}}_i, \mathbf{p}_i$ as shown in Fig. 7(b). The corresponding subdivision rules for deriving new points $[\mathbf{q}, \tilde{\mathbf{q}}]$ from $[\mathbf{p}, \tilde{\mathbf{p}}]$, generalized to account for varying β_i, are

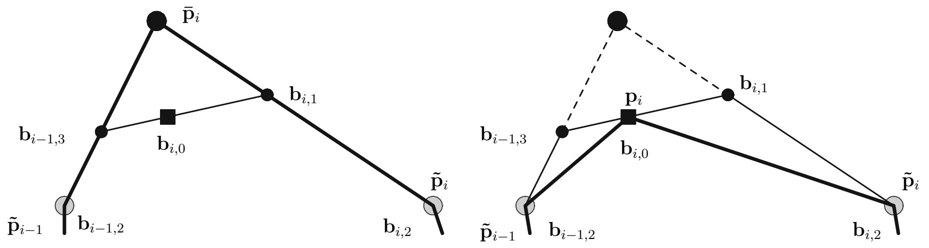


Fig. 9. Control points $\bar{\mathbf{p}}_i, \tilde{\mathbf{p}}_i, \mathbf{p}_i$ and BB-coefficients $\mathbf{b}_{i,k}$, $k = 0 \ldots 4$. (*left*) standard C^2 B-spline to BB-form conversion (12) (*right*) Skip-interpolating conversion with points $\mathbf{p}_i = \mathbf{b}_{i,0}$ on the resulting curve.

$$\begin{aligned} \mathbf{q}_{2i} &:= \mathbf{p}_i, \\ \mathbf{q}_{2i+1} &:= a_0\tilde{\mathbf{p}}_{i-1} + a_1\mathbf{p}_i + a_2\tilde{\mathbf{p}}_i + a_3 p_{i+1} + a_4\tilde{\mathbf{p}}_{i+1}, \\ \tilde{\mathbf{q}}_{2i} &:= e_0\tilde{\mathbf{p}}_{i-1} + e_1\mathbf{p}_i + e_2\tilde{\mathbf{p}}_i, \\ \tilde{\mathbf{q}}_{2i+1} &:= \ell_0\tilde{\mathbf{p}}_i + \ell_1\mathbf{p}_{i+1} + \ell_2\tilde{\mathbf{p}}_{i+1}. \end{aligned} \tag{13}$$

where

$$a_0 := -\frac{1}{8}\frac{\beta_i^2}{\beta_i+1}, \quad a_1 := \frac{1}{8}\beta_i + \frac{3}{16}, \quad a_2 := 1 - a_0 - a_1 - a_3 - a_4,$$
$$e_0 := -\frac{1}{4}\frac{\beta_i^2}{\beta_i+1}, \quad e_1 := \frac{1}{2} + \frac{1}{4}\beta_i, \quad e_2 := 1 - e_0 - e_1,$$

and $[a_3, a_4, \ell_2, \ell_1, \ell_0]$ are obtained from $[a_1, a_0, e_0, e_1, e_2]$ by replacing $\beta_i \to 1/\beta_{i+1}$. This subdivision is called skip-interpolating, since every second point $\mathbf{q}_{2j}$ of the control polygon ends up on the limit curve. The curve inherits curvature continuity in the limit from the underlying C^2 spline.

We note that in both (11) and (13), the discrete curvature of the combined control nets is meaningless: already the subpolygons, $\tilde{\mathbf{p}}_i$ or $\bar{\mathbf{p}}_i$, yield wildly oscillating plots. But the subpolygon based on $\mathbf{p}_i$ shows no spikes, as predicted.

6 Non-uniform Subdivision Based on a Rational Quadratic G^1 Curve Construction

As promised in Section 2, we present a non-uniform subdivision scheme based on a G^1 curve construction. This construction is useful in its own right and its derivation is similar but simpler than that in [14].

Given a control polygon $\mathbf{p}$ and weights ω_i at the control points as in Fig. 10, *middle*, we derive the BB-control-points of a rational quadratic G^1 curve with BB-pieces f_i Fig. 10, *left*, such that numerator and denominator are in BB-form (Bernstein-Bézier form):

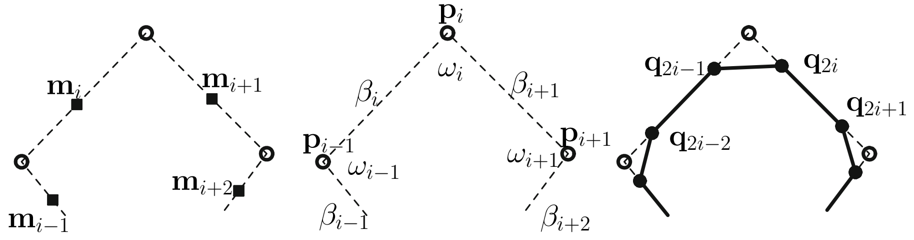


Fig. 10. Construction of Non-uniform Rational Quadratic Subdivision. (*left*) BB-control polygons with end points $\mathbf{m}_i$. (*middle*) Affine control polygon $\mathbf{p}$ with weights ω and the G^1 constant β_i associated with edge $\mathbf{p}_{i-1}\mathbf{p}_i$. (*right*) Once-refined control polygon.

$$f_i : u \mapsto \frac{\sum_{k=0}^{2} w_k \mathbf{b}_k B_k^2(u)}{\sum_{k=0}^{2} w_k B_k^2(u)} \quad B_k^n := \binom{n}{k}(1-u)^{n-k}u^k. \tag{14}$$

In terms of β_i, the BB-control-points of a rational quadratic f_i are

$$\begin{aligned} &\mathbf{m}_i := (1-\nu_i)\mathbf{p}_{i-1} + \nu_i \mathbf{p}_i \ , \qquad \nu_i := \frac{\omega_i}{\beta_i \omega_{i-1} + \omega_i}, \\ &\mathbf{b}_{i,0} := \mathbf{m}_i, \ \mathbf{b}_{i,1} := \mathbf{p}_i, \ \mathbf{b}_{i,2} := \mathbf{m}_{i+1}, \qquad w_{i,0} = w_{i,2} := 1, w_{i,1} := \omega_i. \end{aligned} \tag{15}$$

We now associate the weights ω_i with the coefficients $\mathbf{b}_{i,1}$ and the constants β_i with $\mathbf{m}_i$ and hence with edges $\mathbf{p}_{i-1}\mathbf{p}_i$ (see Fig. 10, *middle*).

To subdivide, we split each quadratic curve segment by de Casteljau's algorithm at its center $u = 1/2$. Then we re-normalize each piece's rational weights so that the first and last are both 1:

$$\begin{aligned} \bar{w}_{i,k} &:= \frac{w_{i,k}}{w_{i,0}} \ , \ k = 0,1,2 \ ; \\ w_{i,k}^{sym} &:= \bar{w}_{i,k} h_i^k, \ k = 0,1,2 \ ; h_i := \frac{1}{\sqrt{\bar{w}_{i,2}}} \ . \end{aligned} \tag{16}$$

Then

$$\beta_i^{sym} := h_{i-1} h_i \beta_i.$$

Subdivision generates the new control points $\mathbf{q}_{2i-1}$, $\mathbf{q}_{2i}$ of the two subquadratics, a symmetrized weight ω_j per point and a symmetrized constant β_j per edge (cf. Fig. 10, *right*) by the following weight-dependent, hence non-stationary, and constant-dependent, hence non-uniform, subdivision algorithm.

Algorithm. [Non-uniform Rational Quadratic Subdivision]
Input: Control polygon $\mathbf{p}$, weights ω and constants β (see Fig. 10, *middle*).
Output: Control polygon $\mathbf{q}$, new weights ω and constants β (see Fig. 10, *right*).

The explicit refinement rules are

$$\mathbf{q}_{2i-1} := a_0^i \mathbf{p}_{i-1} + a_1^i \mathbf{p}_i, \quad \mathbf{q}_{2i} := b_0^i \mathbf{p}_i + b_1^i \mathbf{p}_{i+1}, \tag{17}$$

$$a_0^i := \frac{\omega_{i-1}\beta_i}{(\omega_{i-1}\beta_i + \omega_i)(1+\omega_i)}, \; b_1^i := \frac{\omega_{i+1}}{(\omega_i\beta_{i+1} + \omega_{i+1})(1+\omega_i)}, \tag{18}$$

$$a_1^i := 1 - a_0^i, \qquad b_0^i := 1 - b_1^i. \tag{19}$$

For the next refinement step, set $\dot{w}_i := \sqrt{(1+\omega_i)/2}$ and redefine

$$\mathbf{p}_k := \mathbf{q}_k \;,\; \omega_{2i-1} = \omega_{2i} := \dot{w}_i \;,\; \beta_{2i-1} := \frac{\dot{w}_{i-1}}{\dot{w}_i}\beta_i \;,\; \beta_{2i} := 1 \;.$$

We call the scheme 'non-uniform' to emphasize its dependence on the constants β_i.

The smoothness of the subdivision algorithm is immediate since it corresponds to subdividing an underlying G^1 rational spline. If all weights ω_i are equal and all $\beta_i = 1$ in the algorithm then for $a_0^i = b_1^i = c := \frac{1}{2(1+\omega)}$,

$$\mathbf{q}_{2i-1} := c\mathbf{p}_{i-1} + (1-c)\mathbf{p}_i, \qquad \mathbf{q}_{2i} := (1-c)\mathbf{p}_i + c\mathbf{p}_{i+1}, \quad \omega \leftarrow \sqrt{\frac{1+\omega}{2}}, \tag{20}$$

i.e. the subdivision simplifies to a known uniform non-stationary C^1 circle-reproducing subdivision [9].

Indeed, the non-uniform subdivision can reproduce a number of conics in one framework. Fig. 11(a) illustrates the construction of a circle from an asymmetric circumscribed control polygon. We need only to set

$$w_i := \cos\frac{\alpha_i}{2}, \qquad \beta_i := \frac{\sin\frac{\alpha_i}{2}}{\sin\frac{\alpha_{i-1}}{2}}, \tag{21}$$

where the α_i are the opening angles between consecutive points on the circle that are interpolated by the circumscribed control polygon. Fig. 11 (b) and (c) make the point that the subdivision can reproduce the 'uniform' non-stationary subdivision from [9] but additionally vary shape by varying β_i. The uniform subdivision, even though non-stationary, can only reproduce one primitive at a time, here an ellipse, making designs such as Fig. 1(b) cumbersome. By contrast, Non-uniform Rational Quadratic Subdivision adapts to two or more different prescribed conics by replicating the pieces as rational quadratic splines and converting them to control polygons of the subdivision algorithm. Fig. 11(d) shows (dotted) the circle as in (a), now reproduced from a control polygon that is a circumscribed triangle. For the solid-drawn variant, the weights of the top two control points are increased to locally yield hyperbolic pieces and the β_i has been adjusted to keep the bottom segment exactly on the circle. The curve in (e) corresponds to uniform $\beta_i = 1$.

7 Discussion

Our original goal was to address a shortcoming of recent subdivision algorithms for practical design: none locally reproduces several basic shapes within the same curve by

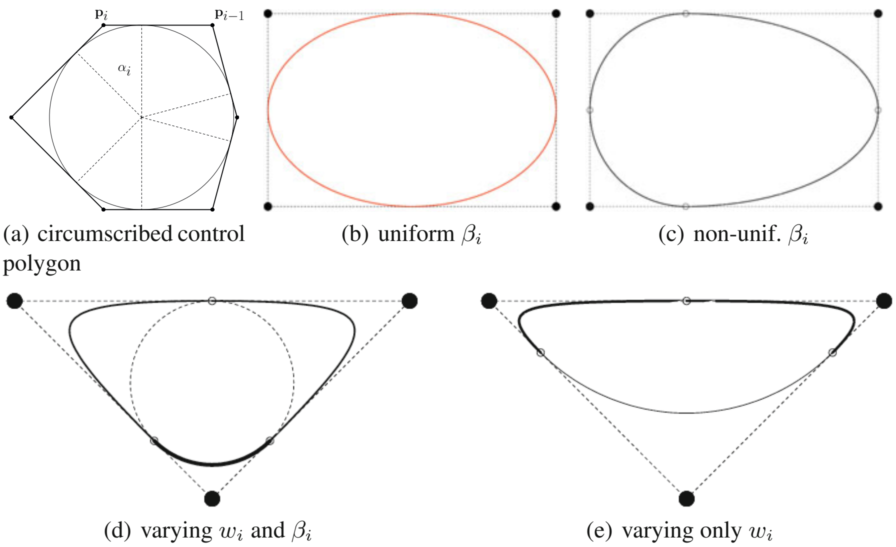


Fig. 11. Non-uniform Rational Quadratic Subdivision. (a) Circle-circumscribed control polygon with unequal opening angles α_i yields a circle (cf. (21)). (b) The result of setting $\beta_i = 1$ and $w_i := cos\frac{\pi}{4}$ is identical to [9]. (c) The result of unequal β_i for $w_i := cos\frac{\pi}{4}$. (d) Circle from a triangle (both *dotted*) and an alternative shape where the circle piece is preserved by varying the β_i corresponding to the bottom; the remainder is replaced by two hyperbolic pieces, abutting at the hollow circle marker, obtained by increasing the top two weights w_i. (e) Same as (d) but with all β_i identical.

one algorithm. In addressing this challenge algorithmically by non-uniform subdivision following the approach of [14], we noticed a second, related challenge, already present in non-uniform subdivision of polynomial splines: While control polygons often work well for extracting first-order information about curves, the experiments in Section 4 and Lemma 1 show that curvature of an approximating non-stationary subdivision is not easily gleaned from control polygons. In retrospect, this should not surprise since control nets without associated generating functions do not allow for a mathematical analysis of the resulting limit shape [16, Introduction].

Among the options that we explored in order to nevertheless generate useful curvature information in a practical way from control polygons, equalization leads to complicated subdivision rules; and selecting subsequences requires additional careful analysis. Skip-interpolation, on the other hand, is a simple technique to be able to read off curvature while still preserving the typically better shape of non-interpolating, approximating subdivision.

Acknowledgments. Work supported in part by NSF Grant CCF-0728797.

References

1. Aspert, N., Ebrahimi, T., Vandergheynst, P.: Non-linear subdivision using local spherical coordinates. Computer Aided Geometric Design 20(3), 165–187 (2003)
2. Augsdörfer, U.H., Dodgson, N.A., Sabin, M.A.: Variations on the four-point subdivision scheme. Computer Aided Geometric Design 27(1), 78–95 (2010)
3. Beccari, C., Casciola, G., Romani, L.: A non-stationary uniform tension controlled interpolating 4-point scheme reproducing conics. Computer Aided Geometric Design 24(1), 1–9 (2007)
4. Beccari, C., Casciola, G., Romani, L.: Shape controlled interpolatory ternary subdivision. Applied Mathematics and Computation 215(3), 916–927 (2009)
5. Cavaretta, A.S., Dahmen, W., Micchelli, C.A.: Stationary subdivision. Mem. Amer. Math. Soc. 93(453), vi+186 (1991)
6. Chalmovianský, P., Jüttler, B.: A non-linear circle-preserving subdivision scheme. Adv. Comput. Math. 27(4), 375–400 (2007)
7. Conti, C., Romani, L.: Affine combination of B-spline subdivision masks and its non-stationary counterparts. BIT Numerical Mathematics 50(2), 269–299 (2010)
8. Deng, C., Wang, G.: Incenter subdivision scheme for curve interpolation. Computer Aided Geometric Design 27(1), 48–59 (2010)
9. Dyn, N., Levin, D.: Subdivision schemes in geometric modelling. Acta Numerica, 73–144 (2002)
10. Dyn, N., Floater, M.S., Hormann, K.: Four-point curve subdivision based on iterated chordal and centripetal parameterizations. Computer Aided Geometric Design 26(3), 279–286 (2009)
11. Farin, G.: Curves and Surfaces for Computer-Aided Geometric Design. Academic Press, London (1988)
12. Grohs, P.: Approximation order from stability for nonlinear subdivision schemes. Journal of Approximation Theory 162(5), 1085–1094 (2010)
13. Hohmeyer, M.E., Barsky, B.A.: Rational continuity: Parametric, geometric, and frenet frame continuity of rational curves. ACM Transactions on Graphics 8(4), 335–359 (1989)
14. Karčiauskas, K., Peters, J.: Rational G^2 splines. Graphical Models, pp. 1–23 (in revision)
15. Morin, G., Warren, J.D., Weimer, H.: A subdivision scheme for surfaces of revolution. Computer Aided Geometric Design 18(5), 483–502 (2001)
16. Peters, J., Reif, U.: Subdivision Surfaces, Geometry and Computing, vol. 3. Springer, New York (2008)
17. Prautzsch, H., Boehm, W., Paluszny, M.: Bézier and B-spline techniques. Springer, Heidelberg (2002)
18. Romani, L.: From approximating subdivision schemes for exponential splines to high-performance interpolating algorithms. Journal of Computational and Applied Mathematics 224(1), 383–396 (2009)
19. Romani, L.: A circle-preserving C^2 hermite interpolatory subdivision scheme with tension control. Computer Aided Geometric Design 27(1), 36–47 (2010)
20. Sabin, M., Dodgson, N.: A circle-preserving variant of the four-point subdivision scheme. In: Mathematical Methods for Curves and Surfaces: Troms 2004, Modern Methods in Mathematics, pp. 275–286. Nashboro Press (2005)
21. Schaefer, S., Vouga, E., Goldman, R.: Nonlinear subdivision through nonlinear averaging. Comput. Aided Geom. Des. 25(3), 162–180 (2008)
22. Schaefer, S., Warren, J.: Exact evaluation of non-polynomial subdivision schemes at rational parameter values. In: Pacific Graphics, pp. 321–330 (2007)

Fitting a Surface to One of Its Sectional Planar Curves Using Adaptive Trees in Spaces of Curves

Yannick L. Kergosien

LIM&BIO, Université Paris 13
196, rue des Rabats
92160 Antony, France
yannick.kergosien@libertysurf.fr

Abstract. We motivate from interventional medical imaging some geometric problems where one should identify how a curve could be generated from a known surface, asking for the pose of the surface from the outline of an orthogonal projection of it, or the position of a sectional plane from an isometric instance of the sectional curve. We describe a distance between curves that is efficiently implementable and use it to build stochastic trees in metric spaces towards subsets of curves of interest that either have strong convergence properties or permit quick searches on curves, and propose a real time application on a simplified version of such problem.

Keywords: metric, search, tree, adaptive, self-organizing, curve, surface, imaging, registration.

1 Introduction

Medical imaging, either for diagnostic or interventional purposes, provides many instances of search problems in spaces of geometric objects such as curves or surfaces. Some of these can be successfully addressed using the methods of differential calculus in extended settings, but we found that some more direct search methods adapted from computer science to metric spaces can also be useful. We first motivate the problem of fitting a given surface to one of its planar sectional curves, then sketch how calculus, when used for such a task, is hindered by difficulties that call for novel approaches. We shall also refer later to the classic problem of inferring the pose of a known polyhedron from the outline of one of its projections, which can be addressed by very similar procedures.

1.1 Image-Volume Fusion for Interventional Imaging and the Retrieval of Surface Sections

When performing an image guided intervention on a patient – for instance a biopsy or a radio-frequency ablation – one frequently has to use an imaging modality which is less informative than what is available in a purely diagnostic setting. For instance, for a patient that already had very informative 3-dimensional CT or MRI scans to image a tumor, it happens that during the intervention one can only use 2-dimensional ultrasound imaging which only allows to interactively control the probe and to choose the sectional plane while monitoring the images displayed in real time. Some lesions however give less contrast in that modality and it is highly desirable to fuse the 2D ultrasonographic data with the available 3D data previously acquired in the other modality. Such fusion has been implemented using tracking devices that measure the position and pose of the probe [1,24]. It is natural to ask whether it could be possible to perform the fusion without such hardware based tracking, using only the 2D image and the 3D data to infer the position and pose of the probe.

In that direction, we shall first simplify the problem to a more geometric one by supposing that the information consists only in the shape of some surface in space (the analog of a contrast surface, extractable from the 3D data), the planar sections of which constitute the content of the 2D sectional images. Of course real images have more content, for instance in the form of gray levels or texture, which can still be used in the cases where our surface-based approach is not sufficient. We further simplify the problem not taking into account possible boundaries of such a surface and considering only the case of a non-convex triangulated polyhedron.

1.2 Pose Identification from Projections for Computer-Assisted Surgery

Computer-Assisted Surgery sometimes makes use of projection imaging to solve the very similar problem of identifying during an intervention the pose of a known object, for instance a bone (Fig. 1) the geometry of which is already known from a former CT acquisition [18,19]. Computer Vision also deals with such questions. We shall apply the same methods for the former problem and a simple version of this one.

2 Retrieving Sections on a Surface and Other Intended Applications

2.1 A Distance between Plane Curves or Plane Curve Shapes

The discrete version of the distance we now define will be used throughout this paper. It is a distance between shapes of plane curves but it also provides ways to compute vector fields along curves, to be used in the calculus approach. Let $\mathbb{S}^1$ be the unit circle parameterized by $t \in [0, 2\pi]$ and $\sigma_1, \sigma_2 : \mathbb{S}^1 \to \mathbb{R}^2$ two smooth mappings from the unit circle to the plane.

To compare two plane curves C_1, C_2 which are the images of $\mathbb{S}^1$ by two smooth mappings, we reparameterize them proportionally to curvilinear abcissa in $\mathbb{R}^2$ thus getting two embeddings $\overline{\sigma_1}$, $\overline{\sigma_2}$ of $\mathbb{S}^1$ in $\mathbb{R}^2$, and compute

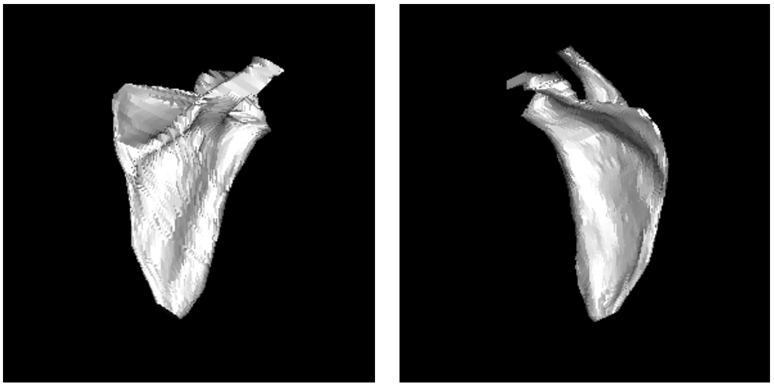


Fig. 1. Rendered views of a human right scapula model reconstructed from CT. The planar sections of this surface will be searched using a tree like the one shown Fig. 8.

$$d_c(C_1, C_2) = \min_{\phi \in [0,2\pi]} \left(\int_0^{2\pi} \|\overline{\sigma_2}(t+\phi) - \overline{\sigma_1}(t)\|^2 dt \right)^{\frac{1}{2}} . \tag{1}$$

Using the value $\hat{\phi}$ of ϕ that minimizes the former expression (cases of non-unicity must be dealt with in algorithms), one can define the barycenter of two curves C_1 and C_2 :

$$(1-\lambda)C_1 \hat{+} \lambda C_2 : t \mapsto (1-\lambda)C_1(t) + \lambda C_2(t+\hat{\phi}) . \tag{2}$$

If we fix a group $\mathcal{G}$ of transformations acting on the elements of the affine plane, either consisting of all isometries or only the direct isometries, we call *shape* of a curve C its orbit under $\mathcal{G}$, and we can compute a distance between the curve shapes c_1, c_2 knowing a representative curve for each, say C_1 and C_2 and setting

$$\begin{aligned} d_s(c_1, c_2) &= \min_{T \in \mathcal{G}} d_c(C_1, T(C_2)) \\ &= \min_{(T,\phi) \in \mathcal{G} \times [0,2\pi]} \left(\int_0^{2\pi} \|T\left(\overline{\sigma_2}(t+\phi)\right) - \overline{\sigma_1}(t)\|^2 dt \right)^{\frac{1}{2}} . \end{aligned} \tag{3}$$

That these formulas for d_c and d_s provide actual distances results from Minkowski's inequality (see below the discrete case). Using the values $\hat{T}$ of T and $\hat{\phi}$ of ϕ that minimize the former expression (cases of non-unicity must be dealt with in algorithms), one can also define the barycenter of two curve shapes c_1, c_2 with respective representative curves C_1 and C_2 : a representative curve of it is

$$(1-\lambda)c_1 \mathring{+} \lambda c_2 : t \mapsto (1-\lambda)C_1(t) + \lambda \hat{T}(C_2(t+\hat{\phi})) . \tag{4}$$

Discrete Case. We now redefine these distances and barycenters for plane polygonal curves (polygons) to be used in applications. We will write a n-gon C

as an ordered set $(C_i)_{i=0}^{n-1}$ of vertices in $\mathbb{R}^2$, and n-gon shapes will still be the orbits of n-gons under the action of a group $\mathcal{G}$ of isometries of the plane. A n-gon will be called *equilateral* if all its sides have equal lengths:

$$\forall i, 0 \leq i \leq n-1, \quad \|C_{i+1 \pmod n} - C_i\| = \|C_1 - C_0\| \; .$$

For each n, we only compare equilateral n-gons, which practically requires fixing n and remapping all the curves of interest, like plane sections of surfaces or barycenters of n-gons (see below), to such equilateral n-gons. If C_1 and C_2 are equilateral n-gons $(C_{1,i})_{i=0}^{n-1}$ and $(C_{2,i})_{i=0}^{n-1}$, we set

$$l(C_1, C_2, p) = \left(\Sigma_{i=0}^{n-1} \|C_{2,\, i+p \pmod n} - C_{1,i}\|^2\right)^{\frac{1}{2}} \; , \tag{5}$$

$$\begin{aligned} d_c(C_1, C_2) &= \min_{k \in \{0,\ldots,n-1\}} l(C_1, C_2, k) \\ &= \min_{k \in \{0,\ldots,n-1\}} \left(\Sigma_{i=0}^{n-1} \|C_{2,\, i+k \pmod n} - C_{1,i}\|^2\right)^{\frac{1}{2}} \; , \end{aligned} \tag{6}$$

and for equilateral n-gon shapes c_1 and c_2 represented by such C_1 and C_2 we define

$$\begin{aligned} d_s(c_1, c_2) &= \min_{T \in \mathcal{G}} d_c(C_1, T(C_2)) \\ &= \min_{k \in \{0,\ldots,n-1\}} \min_{T \in \mathcal{G}} \left(\Sigma_{i=0}^{n-1} \|T\left(C_{2,\, i+k \pmod n}\right) - C_{1,i}\|^2\right)^{\frac{1}{2}} \; . \end{aligned} \tag{7}$$

Let us now prove in this discrete setting that d_c and d_s are distances.

Theorem 1. *Let n be a positive integer, d_c is a distance on the set of equilateral n-gons in $\mathbb{R}^2$ and d_s is a distance on the set of equilateral n-gon shapes in $\mathbb{R}^2$.*

Proof. We first need:

Lemma 1. *Let C_1, C_2, C_3 be n-gons and p, q, r be integer such that $p + q = r \pmod n$, then*

$$l(C_1, C_3, r) \leq l(C_1, C_2, p) + l(C_2, C_3, q) \; . \tag{8}$$

Proof. (Of the Lemma) Minkowski's inequality can be written

$$\begin{aligned} &\left(\Sigma_{i=0}^{n-1} \|C_{3,\, i+r \pmod n} - C_{1,i}\|^2\right)^{\frac{1}{2}} \\ &\leq \left(\Sigma_{i=0}^{n-1} \|C_{2,\, i+p \pmod n} - C_{1,i}\|^2\right)^{\frac{1}{2}} + \left(\Sigma_{i=0}^{n-1} \|C_{3,\, i+q \pmod n} - C_{2,i}\|^2\right)^{\frac{1}{2}} \end{aligned}$$

since hypothesis $p+q = r \pmod n$ implies that for each i vertices $C_{3,i+r \pmod n}$ and $C_{3,i+p+q \pmod n}$ coincide. □

Now for the theorem. For any equilateral n-gons C_1, C_2, C_3, call $\hat{p}$, $\hat{q}$, $\hat{r}$ integers which respectively minimize $l(C_1, C_2, p)$, $l(C_2, C_3, q)$ and $l(C_1, C_3, r)$. The lemma tells us that

$$l(C_1, C_3, \hat{p} + \hat{q}) \leq l(C_1, C_2, \hat{p}) + l(C_2, C_3, \hat{q}) = d_c(C_1, C_2) + d_c(C_2, C_3), \tag{9}$$

and by definition

$$d_c(C_1, C_3) \leq l(C_1, C_3, \hat{p} + \hat{q}) \ . \tag{10}$$

For three n-gon shapes c_1, c_2, c_3, represented by three n-gons C_1, C_2, C_3, call $(\hat{p}, \hat{T})$, $(\hat{q}, \hat{U})$, $(\hat{r}, \hat{V})$ values in $\mathbb{N} \times \mathcal{G}$ which minimize respectively $l(C_1, T(C_2), p)$, $l(C_2, U(C_3), q)$ and $l(C_1, U \circ T(C_3), r)$. By the lemma and the fact that elements of $\mathcal{G}$ are isometries

$$\begin{aligned} l(C_1, U \circ T(C_3), \hat{p} + \hat{q}) &\leq l(C_1, T(C_2), \hat{p}) + l(T(C_2), U \circ T(C_3), \hat{q}) && (11)\\ &= l(C_1, T(C_2), \hat{p}) + l(C_2, U(C_3), \hat{q}) && (12)\\ &= d_s(C_1, C_2) + d_s(C_2, C_3), && (13) \end{aligned}$$

and by definition

$$d_s(C_1, C_3) \leq l(C_1, U \circ T(C_3), \hat{p} + \hat{q}) \ . \tag{14}$$

□

A barycenter of two equilateral n-gons C_1 et C_2 with weights α and β such that $\alpha + \beta \neq 0$ can be defined, using a value $\hat{p}$ that minimizes $l(C_1, C_2, p)$:

$$\alpha C_1 \hat{+} \beta C_2 = (\alpha C_{1,i} + \beta C_{2, i+\hat{p} \pmod n})_{i=0}^{n-1} \ , \tag{15}$$

and for two equilateral n-gon shapes c_1, c_2 with respective representatives equilateral n-gons C_1 and C_2, if $(\hat{p}, \hat{T})$ is a value in $\mathbb{N} \times \mathcal{G}$ which minimizes $l(C_1, T(C_2), p)$, the barycenter of c_1 and c_2 with weights α and β can be defined as the class of

$$\alpha C_1 \mathring{+} \beta C_2 = (\alpha C_{1,i} + \beta C_{2, i+\hat{p} \pmod n})_{i=0}^{n-1} \ . \tag{16}$$

However, in both cases, these expressions are not in general equilateral n-gons and we need to use some approximation to stay in the set of equilateral n-gons. We shall also use these formulas with $\alpha + \beta = 0$ to compute vector fields along curves or curve shapes. In the discrete case a vector field along a curve is simply a mapping which associates a vector to each vertex of the n-gon. With $\alpha = -1$ and $\beta = 1$ we shall note $C_2 \hat{-} C_1$ for $\alpha C_1 \hat{+} \beta C_2$ and and $C_2 \mathring{-} C_1$ for $\alpha C_1 \mathring{+} \beta C_2$.

Remark. The choice of the discrete distance just described was led by computational efficiency. There exist exact formulas to minimize over transforms in $\mathcal{G}$ which have been used in Procrustes' methods of Statistics and are based on singular value decomposition [17,12]. Using n-gons with n of the order of 100 is still tractable on PCs for real-time applications. Inherent parallelism could be further exploited.

2.2 Setting for the Section Retrieval Problem

Consider a surface S in $\mathbb{R}^3$, image of a smooth embedding of the 2-sphere $\mathbb{S}^2$ into $\mathbb{R}^3$. We fix a plane P in $\mathbb{R}^3$, say $\{(x, y, z) \in \mathbb{R}^3 : z = 0\}$, and consider the group

$\mathcal{D}$ of direct affine isometries of $\mathbb{R}^3$ (which can be expressed by a rotation followed by a translation). The problem of section retrieval can be phrased: "given a non empty subset C of P such that there exists $T \in \mathcal{D}$ with $C = T(S) \cap P$, find such a T ". A more general question : "find all elements $T \in \mathcal{D}$ which minimize the distance from C to $T(S) \cap P$", for an adequate distance, permits at the same time to account for imprecise data and to address the decision problem of whether a T exists starting from a more general C. One can address that problem under some extra hypotheses, for instance to avoid the intersection being reduced to a single point, or to suppose that the intersection is transversal.

2.3 Calculus Approach for Section Retrieval

Infinitesimally perturbing T produces changes of $C_{T(S)} = T(S) \cap P$ which can be viewed as vector fields along C (see for instance [7]). Starting from a position $T_0(S)$ of S producing an intersection $T_0(S) \cap P$ with P, if we could find a path in $\mathcal{G}$ which would progressively deform the "source curve" $T_0(S) \cap P$ into the "target curve" $C_t = T(S) \cap P$, we would eventually get T. Several difficulties arise, however. First, if the surface is not convex, the topology of the intersection set can change, at least requiring the use of an adequate distance to compare non homeomorphic sets. Second, even if no such change happens – allowing us to use the distances formerly described – the mapping from infinitesimal transforms on $T(S)$ to vector fields along $T(S) \cap P$, despite linear, is not invertible.

2.4 Implementation

We suppose S to be a (non necessarily convex) known polyhedron in $\mathbb{R}^3$ and C_t, the *target curve*, to be a plane section of it given by an isometric instance of it in P. More precisely, C_t is a given polygonal curve in P such that there exists $T_t \in \mathcal{D}$ with $C_t = T_t(S) \cap P$, where T_t is an unknown transform which we want to compute from C_t.

In the whole implementation [12] we only deal with transforms T of S that equivalently lead to sections of S by planes $T^{-1}(P)$ that are in general position, i.e., not containing any vertex of S (which also guarantees that intersections of edges of $T(S)$ by P are transversal). This condition is easy to check and it simplifies the algorithmics of intersection computing, especially the topological types of the intersection sets. If not met, we slightly perturb the plane to get it, but this is extremely rarely necessary. The effect of a perturbation of the plane in such condition depends only on the set of vertices of the edges that are intersected, which we call a *tube*. We call $C_{T(S)}$ the intersection of $T(S)$ and P.

Infinitesimal motions that act on S are elements of the tangent space at the identity to the group $\mathcal{D}$ of displacements in $\mathbb{R}^3$ ($\mathcal{D}$ is a semi-direct product of SO(3) for rotations and $\mathbb{R}^3$ for translations), i.e. elements of its Lie algebra. The Jacobian of the mapping h from elements of $\mathcal{D}$ to their corresponding $C_{T(S)}$ can be computed as a composition of a linear mapping Tf from the Lie algebra $T\mathcal{D}$ to the tangent space of the manifold of tubes and a linear mapping Tg from the tangent manifold of tubes to the vector space of vector fields along $C_{T(S)}$.

Its matrix J_h can be written and for any desired deformation V of $C_{T(S)}$, i.e., for any vector field along $C_{T(S)}$ – expressed as a column matrix –, one can find a generalized solution to the equation $J_h X = V$ to get a desired infinitesimal transformation X. There is then an exact formula for the exponential – from the Lie algebra of $\mathcal{D}$ to $\mathcal{D}$ – which from X permits to compute a finite transformation in $\mathcal{D}$ (see [12] for full details). The desired vector field V along C_1 is computed as $V = \epsilon(C_t \hat{-} C_{T(S)})$ with ϵ a positive constant. One can also use $V = \epsilon(C_t \overset{\circ}{-} C_{T(S)})$, which may avoid some of the local minima.

Starting from a certain value of T, and knowing S and C_t, one thus iterates the sequence:

1. compute and parametrize $C_{T(S)} = T(S) \cap P$
2. compute matrices J_h and $V = \epsilon(C_t \hat{-} C_{T(S)})$ (or $V = \epsilon(C_t \overset{\circ}{-} C_{T(S)})$)
3. find a generalized solution to $J_h X = V$
4. compute a finite displacement $\exp X$ and apply it to $T(S)$

until some threshold is reached for the distance $d_c(C_{T(S)}, C_t)$.

2.5 The Idea of an Atlas to Initialize Searches

We implemented this scheme with anatomical surfaces such as a human scapula (Fig. 1) – reconstructed from CT data – which have very diverse sectional shapes. The convergence is satisfactory when the initial position is close enough to the position to be found, however the evolution often terminates in local minima of the distance to the target curve, and novel approaches have to be found because classical remedies like simulated annealing of multiresolution techniques seem to require too much computer time and human supervision for the intended applications. As a first obvious remedy, we used an “atlas” of sectional curves to start from which were recorded with the corresponding positions of the surface, for instance constructed from some uniform sampling of the possible sectional planes, which notably improved the results. Exploring the possibility of doing as much search as possible with such a brute force approach on sets of sectional curves lead to investigate the problems of building the database and achieving good performance of the queries. There is a vast literature about searches in databases of high dimensional objects [22] that can apply to the present problem. We present here an approach that was derived from self-organization research [16,10] and turns out to be quite versatile.

2.6 A Similar Problem in Computer Vision: Finding the Pose of a Known Solid from Its Projection Outline

For this problem, we consider a polyhedral surface free to move around one of its points, and for each of its poses we compute its orthogonal projection on a fixed plane, only keeping the outline of the projection (Fig. 2) to answer the question : “knowing the surface and the outline curve, retrieve the pose of the surface”. We shall take as the surface a polyhedron $S \subset \mathbb{R}^3$, non necessarily convex, and call *outline* of its image by a projection π the frontier of the non compact connected component of $\mathbb{R}^3 \setminus \pi(S)$, which keeps the topological type of a circle, most conveniently for the distance we intend to use. Here the unknown is a rotation matrix, i.e., the *pose* of the surface. We shall use for this problem a pure search approach and build an "atlas" or database of contour outlines with the records of their corresponding rotation matrices, not considering further refinement by interpolation or descent on distance. Moreover, as we shall see, it is advantageous to work on curve shapes rather than curves as the problem is invariant by rotations in the plane of projection.

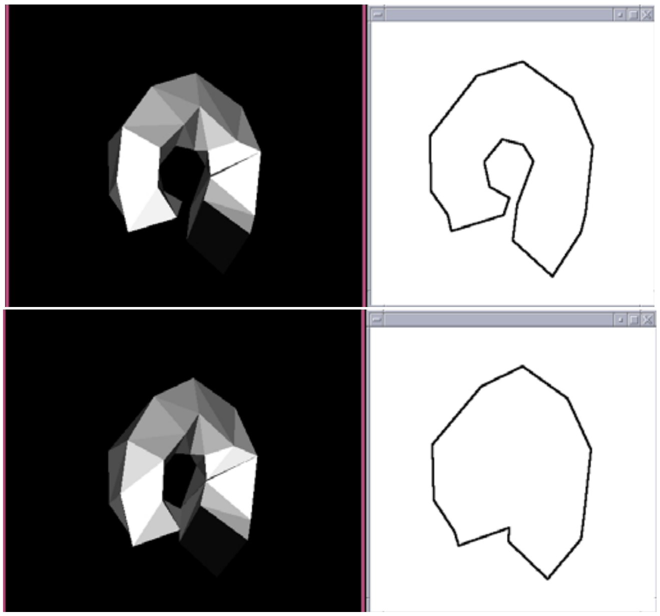


Fig. 2. A polyhedral surface and the outlines of its projections for two different poses

3 Adaptive Trees: Algorithms and Convergence Results

Adaptive trees [13,14] originated in self-organization research, both in the context of neural networks[10] and the morphogenesis of branching phenomena [11]. They share some features with Self-Organizing Maps (SOM) [16] and also Classification And Regression Trees (CART) [3]. They consist in algorithms which grow trees in metric spaces with some convergence to given target subsets of these metric spaces. In the present study, the process will take place in the space of plane curves and will build a tree of curves progressively approximating the different curves contained in the target, the target being either the set of plane sections of the surface or the set of outlines of projections of the surface. A variant of the algorithm will be later introduced to build adaptive *search* trees to address our problems.

3.1 Adaptive Trees

Let (E, d) be a metric space with distance d, and T a subset of it, $T \subset E$, called the *target*, which is the support of a probability measure P defined on E. We are going to evolve in time a *growing set* G_t, where for each $t \in \mathbb{N}$, $G_t \subset E$, starting from a given finite subset $G_0 \subset E$ which we call the *seed* (a single point if we want to get a tree). At each step of the algorithm, G_t is built from G_{t-1} adding to it a point – we call this operation an *accretion* to G_{t-1} – which depends on a point randomly selected from the target T using probability P. A pseudo-code for the algorithm used is:

Adaptive Tree Algorithm

```
1. set $t = 0$
2. until some condition on $t$ or $G_t$ is met, repeat:
   (a) increment $t$
   (b) randomly select a point $c_t$ from $T$ using $P$
   (c) compute the point $a_t$ in $G_{t-1}$ which minimises $d(c_t, a_t)$, possibly using
       a randomised rule to break ties
   (d) set $b_t = (1 - \epsilon)a_t + \epsilon c_t$ and $G_t = G_{t-1} \cup \{b_t\}$, call $b_t$ a child of $a_t$
```

Line d) in the loop is an accretion which also builds the tree structure. We say that the accretion of b_t took place at a_t. It uses a barycenter construction but the algorithm can be also used in metric spaces, such as Riemannian manifolds, where it is possible to find b_t aligned with a_t and c_t, i.e. such that $d(a_t, b_t) + d(b_t, c_t) = d(a_t, c_t)$, with $d(a_t, b_t) = \epsilon d(a_t, c_t)$. The parameter ϵ can be chosen in a wide range of values (say from 10^{-1} to 10^{-4}) and makes the tree more regular if small.

Running the algorithm in the plane, from a single point towards a line segment, as well as in higher dimension (Fig. 3) shows how G_t progressively develops branchings with most of its leaves approaching the target set. Growing G_t in the plane from a single point towards more complex targets such as the one obtained by mixing the probability laws of uniform distributions on a circle and a disk (Fig. 4) exemplifies further some kind of adaptivity property. Please notice that in the process some small branches stop growing. This phenomenon is known as *abortive branching* and is studied in [10,15].

In order to state a convergence theorem, we need some definitions. A point in E is called *active at t* if it belongs to G_t and has a positive probability of being selected as a_t. For any c in T we define *the set of points active for c at time t* to be $\Gamma_t(c) = \{a \in G_t |\ d(c, a) = d(c, G_t)\}$. The *target share* of a point $a \in G_t$ is the set $S_a = \{c \in T |\ \forall a' \in G_t,\ d(c, a) \leq d(c, a')\}$. An element of G_t with a void or P-negligible target share is obviously inactive and will remain so since the other points of G_t which are closer to some points of the target will also be in $G_{t'}$ for all $t' > t$.

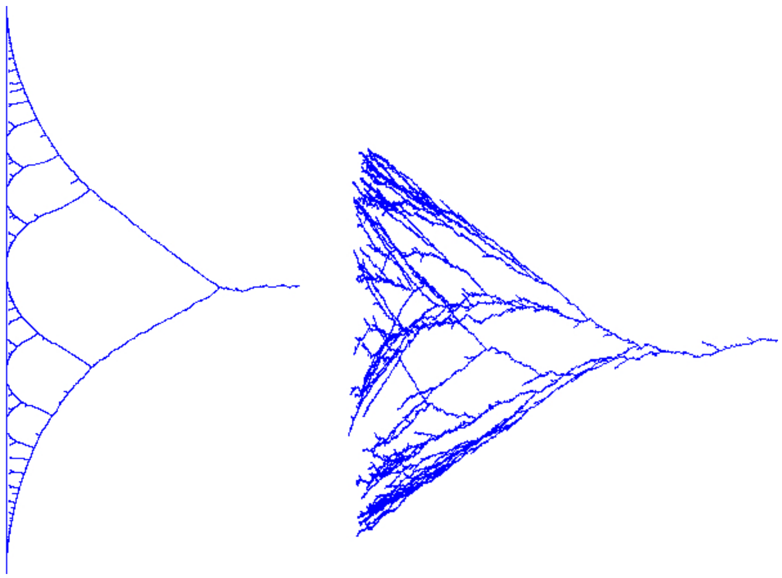


Fig. 3. Adaptive trees. Left: tree grown towards a line segment in $\mathbb{R}^2$. Right: tree grown towards a 7-hypercube in $\mathbb{R}^8$, projected on a plane normal to the target.

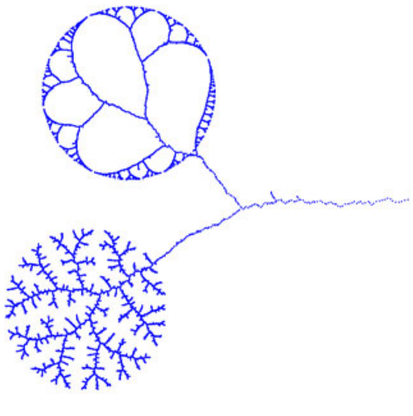


Fig. 4. Adaptive tree grown in $\mathbb{R}^2$ towards a target mixing the probability laws of a circle with uniform distribution (top) and a disk with uniform distribution (bottom)

We can then state:

Theorem 2 ([15]). *Let (E, d) be a metric space, T a compact subset of E which is the support of a probability measure on E, and G_0 a finite subset of E. The following property is P-almost sure: for any real $\xi > 0$ there exists a finite time t from which the sets $G_{t'}$, $t' \geq t$ grown from G_0 using the adaptive tree algorithm will all satisfy the two properties*

1. *for any $c \in T$ there exists $x \in G_{t'}$ such that $d(c, x) < \xi$, i.e. no point of the target is more than ξ away from $G_{t'}$*
2. *for any $x \in G_{t'}$ and any $c \in T$, if x is active for c at t then $d(x, c) < \xi$, i.e. active points are no farther than ξ away from any point of their target share.*

Proof. A full proof can be found in [15].

Remark 1. Theorem 2 obviously implies the following more concise result:

Theorem 3. *Let (E, d) be a metric space, T a compact subset of E which is the support of a probability measure on E, and G_0 a finite subset of E. Almost surely the set of active points at time t converges to T in the Hausdorff metric associated to d.*

3.2 Adaptive Search Trees

The former algorithm has no direct use for a search. A variant of it that we now describe permitted searches in some real time applications. Its convergence is however not guaranteed, as the tree growth, despite using very similar rules, does not take place in a metric space. It builds a binary search tree which is used at each step to quickly retrieve the active point in G_t which is closest to the c_t selected from the target, with considerable time improvement. The idea, quite natural for search trees, is to split the target when a branching occurs – using an oblique hyperplane as a separatrix – and to permanently assign each part of it to one of the two subtrees to be grown from the branching point – each target share assigned to the closest child of the branching point – then recursively nesting successive splits. The permanent character of each of the target splits brings at the same time computational efficiency and the risk of less adaptivity to the target.

We shall assume G_0 to be a singleton, and we shall say that a *branching* occured at time t if the point a_t where the accretion of b_t took place to build G_t already had a child in G_{t-1}; a_t is then called a *branching point*. To make the new algorithm comparable to the former one, we use instead of d a function $d_n : T \times E \mapsto \mathbb{R}^+$, starting with $d_1 = d$, which is updated after each branching and does not remain a distance.

Adaptive Search Tree Algorithm

```
1. set $t = 0$, $d_1 = d$
2. until some condition on $t$ or $G_t$ is met, repeat:
   (a) increment $t$
   (b) randomly select a point $c_t$ from $T$ using $P$
   (c) compute the point $a_t$ in $G_{t-1}$ which minimises $d_t(c_t, a_t)$, possibly using
       a randomised rule to break ties
   (d) set $b_t = (1 - \epsilon)a_t + \epsilon c_t$ and $G_t = G_{t-1} \cup \{b_t\}$, call $b_t$ a child of $a_t$
```

(e) if no branching occured at a_t, set $d_{t+1} = d_t$, otherwise update d_{t+1} with the following rule: let b', b'' the two children of a_t (one of them is b_t). For any x in T such that $d_t(x, b') < d_t(x, b'')$ and any y in E, set $d_{t+1}(x, y) = d_t(x, y)$ if $d_t(y, b') < d_t(y, b'')$, otherwise set $d_{t+1}(x, y) = \infty$. For x in T such that $d_t(x, b'') < d_t(x, b')$, apply the same rule swapping the roles of b_1 and b_2. Test for ties and provide a randomized rule if a tie happens.

When a point c_t is selected from T, in order to find the a_t in G_t which minimizes $d_t(c_t, a_t)$, one successively compares the values of $d(c_t, b'_1)$ and $d(c_t, b''_1)$ to choose a subtree of G_t, where b'_1 and b''_1 are the two children of the branching point closest to the root, then $d(c_t, b'_2)$ and $d(c_t, b''_2)$ are compared to choose the next subtree, where b'_2 and b''_2 are the children of the first branching point in the subtree already chosen, and so on, as in simple binary search trees.

To speed up searches it is efficient to maintain, besides the data structure holding all the accretion points of the tree, a separate tree data structure, which we call the *reduced tree*, obtained by removing from G_t all the non-branching points from which one cannot reach a leaf without crossing a branching point. We call *nodes* the vertices of the reduced tree which come from branching points of G_t and we store with each node ν_k coming from a point p_k in G_t the two children of p_k in G_t (or references to them). We shall call *restricted tree* the restriction of the reduced tree to the nodes.

Experimentally (Fig. 5), the trees resulting from this algorithm are very similar to those produced by the former algorithm with the same targets, except for some details, like the absence of abortive branchings. They are produced very quickly and allow fast searches once the tree is built. However, convergence is not guaranteed and some special target shapes – especially when the tree has to cross some parts of the target to reach other parts of it, which might be less of a problem in greater dimensions – show that the adaptive capabilities of the trees are weaker [14,15].

3.3 Uninformed Accretion Adaptive Trees

The Adaptive Tree Algorithm uses an accretion rule that one could want to avoid. First, it relies on a barycenter construction that, despite taking the form of a pairing efficiently implementable in the case of curves, can be problematic to generalize, e.g., to grow trees in a space of surfaces. Second, when interpreted in the context of Biological Evolution, it infringes the Darwinian paradigm [5] [15] because its accretion step uses some information about the point drawn from the target to construct the point to be accreted. We described an *uninformed adaptive tree* algorithm [15] where the accretion step consists in a random selection of a point b_t in E on a sphere centered at a_t, and for which there is also a theorem of convergence of the set of active points to the target set [15]. This algorithm is compatible with distances, like Hausdorff [4,20] or Gromov-Hausdorff [8,4] distances, which involve no point pairing, and although it is probably less efficient, it permits easier generalizations to new problems.

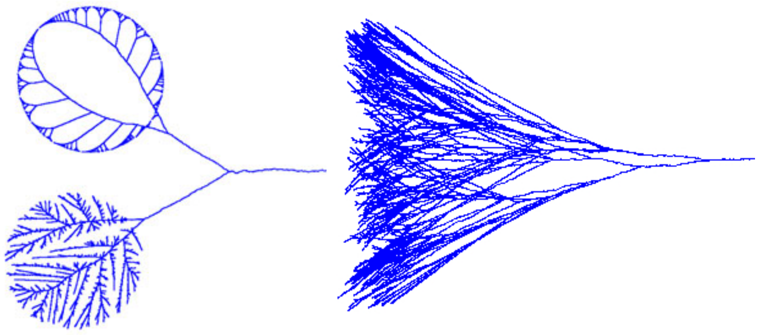


Fig. 5. Adaptive search trees. Left : towards the same target as in Fig. 3. Right : towards a 6-hypercube in $\mathbb{R}^7$, projection of the tree on a plane normal to the target.

4 Using Adaptive Search Trees

4.1 The Pose from Outline of Projection Problem

We used the equilateral n-gon shape distance to grow an adaptive search tree in the space of n-gons towards a target consisting in the set of outlines of a known polyhedron after these outlines were mapped to equilateral n-gons, with a uniform probability measure on the set of 3-dimensional rotation matrices. We fixed $n = 100$ and a number N of steps, then repeated N times a sequence of : (1) randomly selecting a rotation matrix and rotating the polyhedron accordingly, (2) computing the orthogonal projection on a fixed plane of the resulting polyhedron and extracting the outline of that projection, (3) mapping the outline on an equilateral n-gon, (4) accreting a new point to the adaptive search tree started from an equilateral n-gon approximation of an ellipsoid. To solve the pose problem, with each accreted point we recorded, besides the n-gon constructed by the barycentric accretion formula, two matrices : the matrix of the rotation that produced the projection outline, and the matrix of the plane transformation found whle computing the distance between shapes.

Fig. 6 shows the nodes of the restricted tree produced were each node is positioned at a conventional location (like for an abstract binary tree, but with progressive superimposition) and replaced by a representative of its n-gon shape. Observe the progressive differentiation from the ellipsoidal shape of the root towards the diverse shapes of the leaves that approximate the shapes of actual outlines of projections of the surface.

Once a tree built, we could use it in a computer-simulated recognition of the pose from the outline (Fig. 7 is a screen shot of the interface used at the conference) : a 3-dimensional orthogonal shaded view of the polyhedron was displayed on the screen with the possibility to interactively rotate it using a

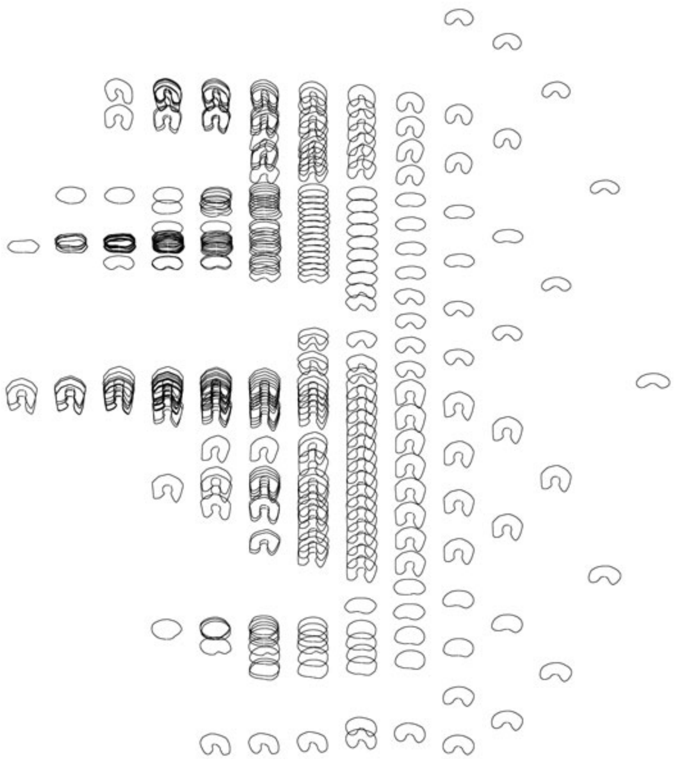


Fig. 6. (Rotated) Nodes of the restricted tree associated to an adaptive search tree grown in the space of 100-gons, starting from the equilateral 100-gon approximation of an ellipsoid, towards the set of equilateral 100-gon approximations of the outlines of the surface shown Fig. 2, after 2000 accretions ($\epsilon = 0.069$). Only the curves associated to the nodes (branching points of the adaptive search tree) are drawn. Notice the progressive differentiation towards actual outlines.

mouse. Simultaneously, the outline of that view was computed and the tree was searched for the closest n-gon shape in the tree. From the two matrices stored in the tree with the retrieved outline shape and the matrix of the 2D transform found while computing the distance between the computed outline (i.e., the query) and the closest tree node (i.e., the answer), the pose was inferred and used to display in a second window a view of the surface in that pose. The fit of the original outline (computed from the surface controlled with the mouse) with the inferred one (computed from the surface displayed with the inferred pose) was visually assessed by superimposition in a third window. The searches were quick enough, on a portable PC, to enable real time testing of the ability to retrieve the pose from the outline.

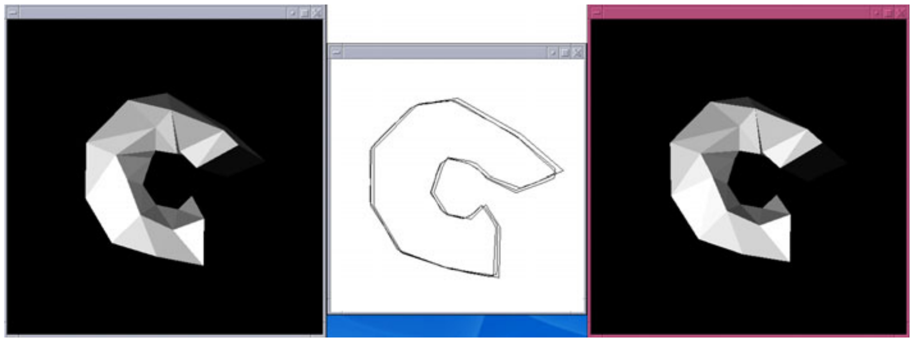


Fig. 7. Interface to check the pose inferred in real-time from projection outlines. The pose of the 3D surface in the left window is interactively controlled by the user; its projection outline is extracted and displayed in real time in the central window. The right window shows in real time the surface in the pose inferred in real time from the projection outline just extracted and its projection outline is superimposed in the central window together with the curve associated to the node returned by the search tree.

4.2 The Plane from Sectional Curve Problem

More control parameters occur in the section problem, which makes an interactive demonstration more difficult to set up. It is however possible to test the possibility of building and using an adaptive search tree for the sections of a surface. We grew an adaptive search tree as before in the space of n-gons, with $n = 100$, towards the set of sections of the surface of a human scapular bone reconstructed from CT data (Fig. 1), using a uniform probability on the set of planes which intersect the surface, and starting from an equilateral n-gon approximation of an ellipsoid. The adaptive tree and adaptive search tree algorithms apparently produced similar results, the search tree version however being faster and enabling quick searches among the sectional curves. As some curves can be non-connected, in cases of several connected components in a sectional curve we kept only the largest connected component and we returned no result for the queries on very small sections.

To be able to use the tree to retrieve a transform which makes the polyhedron intersect the reference plane along the given intersection polygon (up to some n-gon approximations), one can modify the algorithm used to retrieve poses from outlines, starting from an equilateral n-gon approximation of an ellipsoid and iterating N times a sequence of : (1) randomly selecting a 3D displacement 4×4 matrix and transforming the polyhedron accordingly, (2) computing the intersection with the fixed plane of the resulting polyhedron, (3) mapping the intersection on an equilateral n-gon, (4) accreting a new point to the adaptive search tree. With each accreted point we now record, besides the n-gon constructed by the barycentric accretion formula, two matrices : the matrix of the 3D displacement that produced the intersection curve used as the current target point, and the matrix of

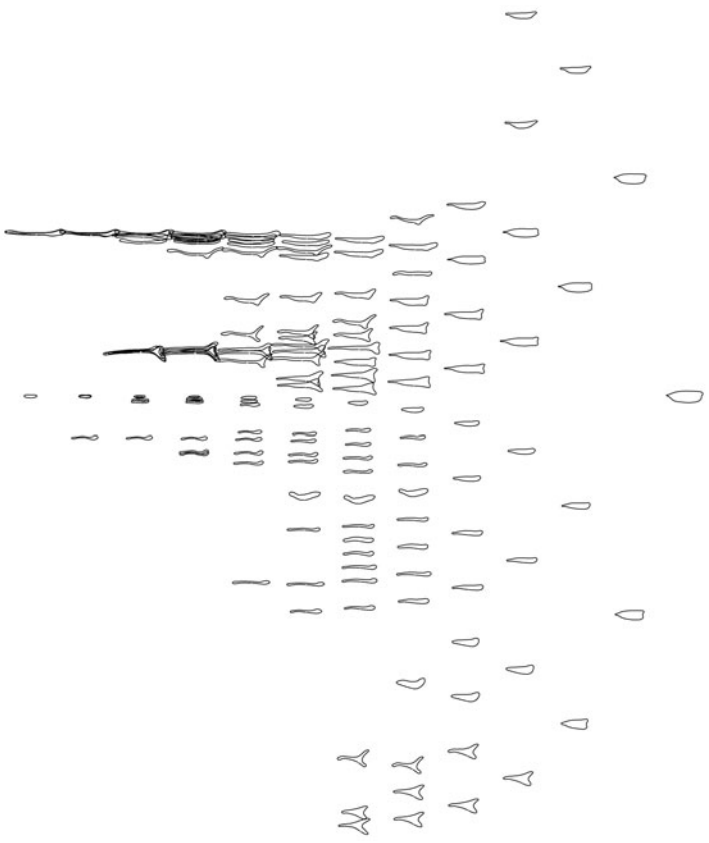


Fig. 8. (Rotated) Restricted tree associated to an adaptive search tree grown in the space of 100-gons, starting from the equilateral 100-gon approximation of an ellipsoid, towards the set of equilateral 100-gon approximations of the plane sections of the surface shown Fig. 1, after 1000 accretions ($\epsilon = 0.09$). Only the curves associated to the nodes (branching points of the adaptive search tree) are drawn. Notice the progressive differentiation towards actual sections of a human right scapula. With each node, two curves slightly different from the one shown here are stored in the data structure, and, given a query sectional curve, the leaf closest to it is found like in a binary search tree, starting from the root and recursively computing shape distances from the query curve to the two curves of the node to decide, depending on which one is less, which child to explore next.

the plane transformation found while computing the distance between the target section and the point where the accretion takes place.

Fig. 8 shows the nodes of the restricted tree produced from the adaptive search tree obtained. Each node is positioned at the conventional location of a binary tree and replaced by a representative of the n-gon shape of the associated point. Observe the progressive differentiation from the ellipsoidal shape of the root towards the diverse shapes of the leaves that approximate the shapes of actual sections of the surface.

Experimental data on the time complexity of the algorithm are reproduced in tables 1 and 2. The algorithm was implemented in language C (but with no multi-thread programming, to be added in the future) with calls to the BLAS level 3 and LINPACK's SVD routine [6], and executed on a Linux system with a 2.4 GHz quad-core processor, for sections of the surface shown Fig. 1, which is a non-convex polyhedron with 2971 vertices and 5764 triangles reconstructed from CT data.

Table 1 gives, for different numbers of accretions, and using 100-sided polygons, the time for growing the tree (in seconds), the number of branching points in the tree (producing as many nodes in the restricted tree), the number of leaves, the mean height of leaves in the restricted tree, and the mean time to answer a curve query once the search tree is available. Typically the tree would be built off-line and only the query time would be critical for the feasibility of a real-time application like the one already shown for projections. The data appear to be compatible with such a possibility. The algorithm being stochastic, these times vary from one instance of the tree to another. For the case of $N = 1000$ accretions the numbers given in the corresponding column are the means and standard deviations measured on a sample of 100 trees. Other columns report a single experiment for each N. In the last row, for each adaptive search tree built, 10000 random sections were used to estimate the mean time (in milliseconds) to retrieve a random section (except for N=1000 : 1000 searches for each of the 100 trees). Notice that this time is approximately linear in the mean height of leaves in the restricted tree.

Table 1. Retrieving sectional curves with an adaptive search tree: execution times depending on the number N of accretion steps, using 100-gons. In the column of 1000 accretions, the numbers are means and standard deviations on a sample of 100 different random trees (each one obtained with 1000 accretions). In other columns, the numbers refer to a single random tree, and thus would vary slightly from tree to tree. In the last row, each mean time for a section search was computed averaging the search times for 10000 random sections, except for N=1000 : 1000 queries for each of the 100 trees.

Number of accretions	1000	4000	16000	64000	256000
Time to grow the tree (s)	7.05 ± 0.43	35	185	1021	5418
Number of branching points	134.52 ± 7.21	620	3409	16890	75662
Number of leaves	74.59 ± 5.62	333	2004	9899	43810
Mean height of leaves in the restricted tree	7.22 ± 0.36	9.73	13.94	19.43	26.24
Mean time for a section search (ms)	8.72 ± 0.49	10.6	14.0	19.5	25.7

Table 2 gives execution times for a fixed number (N=1000) of accretions with increasing numbers n of polygon vertices, which appear to set the limits of real-time feasibility for the present algorithm. Mean search times (in milliseconds) were estimated from 1000 random sections of the surface for each tree.

Table 2. Retrieving sectional curves with an adaptive search tree: execution time depending on the number of vertices of the polygons used for computing the distance; the number of accretions is fixed to 1000

Number of polygon vertices	50	100	200	400	800
Time to grow the tree (s)	3	6	20	66	233
Mean height of leaves	7.39	6.77	7.21	6.80	6.62
Mean time for a section search (ms)	3	8	24	78	275

5 Conclusion and Prospects

After motivating a search approach to some geometric problems arising in interventional imaging medical applications, we described an algorithm which grows trees in metric spaces which can be those of geometric objects if adequate distances are available. We described such a distance to permit searches of plane curves or their approximations by equilateral n-gons. It involves some plane rigid matching and a search for a best parametrization but still can be efficiently implemented numerically. The adaptive tree algorithm has strong stochastic convergence properties to compact subsets on metric spaces where a barycenter can be used or generalized. A variant of it, still with some strong stochastic convergence properties, the so-called non-informed accretion adaptive tree algorithm, allows for distances that do not provide matching, such as Hausdorff or Gromov-Hausdorff distances. An algorithm to produce *search* trees with very similar behaviour towards targets in metric spaces was then described and experimentally tested but it has no guaranteed convergence. It was used to address the two problems described and showed satisfying speed permitting real-time searches in spaces of plane curves.

Several issues are still calling for further research. Extending the distance to more complex objects while keeping computational efficiency is one of them. The search algorithm needs improvement on the side of convergence. Among possible directions, one might try to combine the provably convergent adaptive trees with classical search tree algorithms, using the points newly built. Adaptive trees, despite being built stochastically so far, could also be interesting as descriptors for sets of curves, somewhat like medial axes [2] or Reeb graphs [23] for embedded manifolds. The kind of progressive shape interpolation that they perform, and its analogy with multiresolution, also calls for some exploration of how they could be used for shape matching and morphing. We shall address unicity issues and some related geometric results in a separate publication, of a less applied type.

The practical problem of unicity deserves going back to the original problem where more information is usually available to disambiguate the results, and some useful forms of answer can be sought even in the absence of unicity.

References

1. Barratt, D.C., Davies, A.H., Hughes, A.D., Thom, S.A., Humphries, K.N.: Optimisation and Evaluation of an Electromagnetic Tracking Device for High-Accuracy Three-Dimensional Ultrasound Imaging of the Cartoid Arteries. Ultrasound in Medicine and Biology 27(7), 957–968 (2001)
2. Blum, H.: A transformation for extracting new descriptors of shape. In: Proceedings of Models for the Perception of Speech and Visual Form, pp. 362–380. MIT Press, Cambridge (1967)
3. Breiman, L., Friedman, J.H., Ohlsen, R.A., Stone, C.J.: Classification and regression trees. Wadsworth, Belmont (1984)
4. Burago, D., Burago, Y., Ivanov, S.: A Course in Metric Geometry. Graduate Studies in Mathematics, vol. 33. AMS, Providence (2001)
5. Darwin, C.: On the origin of species by means of natural selection, or the preservation of favored races in the struggle for life. John Murray, London (1859)
6. Dongarra, J., Bunch, J., Moler, C., Stewart, G.W.: LINPACK Users Guide. SIAM, Philadelphia (1979)
7. Golubitsky, M., Guillemin, V.: Stable Mappings ans their Singularities. Springer, New York (1973)
8. Gromov, M.: Metric Structures for Riemannian and Non-Riemannian Spaces. Birkhauser, Boston (1999)
9. Kendall, D.G.: Shape manifolds, Procrustean metrics, and complex projective spaces. Bull. London Math. Soc. 16, 81–121 (1984)
10. Kergosien, Y.L.: Adaptive ramification and abortive concepts. In: Personnaz, L., Dreyfus, G. (eds.) Neural networks from models to applications, Proc. NEURO 1988, June 6-9, pp. 439–449. I.D.S.E.T., Paris (1988)
11. Kergosien, Y.L.: Adaptive ramification: Comparing models for biological, economical, and conceptual organization. Acta Biotheoretica 38, 243–255 (1990)
12. Kergosien, Y.L.: Matching a surface to a given sectional curve. C. R. Acad. Sciences Paris, Biologies 325, 355–365 (2002)
13. Kergosien, Y.L.: Adaptive branching in Epigenesis and Evolution. C. R. Biologies 326, 477–485 (2003)
14. Kergosien, Y.L.: Adaptive trees and pose identification from external contours of polyhedra. In: Olsen, O.F., Florack, L.M.J., Kuijper, A. (eds.) DSSCV 2005. LNCS, vol. 3753, pp. 157–168. Springer, Heidelberg (2005)
15. Kergosien, Y.L.: Distance-driven adaptive trees in biologic metric spaces: uninformed accretion does not prevent convergence. Phil. Trans. R. Soc. A 367, 4967–4986 (2009)
16. Kohonen, T.: Self-organizing maps, 2nd edn. Springer, Berlin (1997)
17. Kraznowski, W.J.: Principles of multivariate analysis. Oxford Univ. Press, Oxford (1988)
18. Lavallee, S.: Registration for computer-integrated surgery: Methodology, state of the art. In: Taylor, et al. (eds.) Computer-Integrated Surgery, ch. 5, pp. 77–97. MIT Press, Cambridge (1995)

19. Lavallee, S., Szeliski, R., Brunie, L.: Anatomy-based registration of three-dimensional medical images, range images, X-ray projections, and three-dimensional models using octree splines. In: Taylor, R.H., Lavallee, S., Burdea, G., Mosges, R. (eds.) Computer Integrated Surgery, pp. 115–143. MIT Press, Cambridge (1996)
20. Molchanov, I.: Theory of Random Sets. Springer, London (2005)
21. Parthasarathy, K.R.: Probability measures on metric spaces. Academic Press, New York (1967)
22. Samet, H.: Foundations of multidimensional and metric data structures. Morgan Kaufmann, San Francisco (2006)
23. Shinagawa, Y., Kunii, T., Kergosien, Y.: Surface coding based on Morse Theory. IEEE Comp. Graph. Appl. 11, 66–78 (1991)
24. Xu, S., Kruecker, J., Turkbey, B., Glossop, N., Singh, A.K., Choyke, P., Pinto, P., Wood, B.J.: Real-time MRI-TRUS fusion for guidance of targeted prostate biopsies. Comput. Aided Surg. 13(5), 255–264 (2008)

Verified Spatial Subdivision of Implicit Objects Using Implicit Linear Interval Estimations

Stefan Kiel

University of Duisburg-Essen
Lotharstr, 63
D - 47057 Duisburg
kiel@inf.uni-due.de

Abstract. In this paper we describe the LIETree, a new data structure for verified spatial decomposition of implicit objects. The LIETree is capable of utilizing implicit linear interval estimations for calculating a verified enclosure of the implicit function's codomain. Furthermore, it uses consistency techniques to tighten the object enclosure. Overall, it delivers improved accuracy and uses fewer nodes than common uniform subdivision schemes using interval or affine arithmetic for enclosure.

1 Introduction

There are various techniques for geometric modeling available, like polyhedral meshes, free form surfaces or implicit objects. In this paper we will only consider objects O described by a single implicit function $f : \mathbb{R}^n \to \mathbb{R}$. A point $x \in \mathbb{R}^n$ belongs to the object if $f(x) \leq 0$. Formally, we have:

$$f(x) = \begin{cases} < 0 & x \in O \setminus \partial O \\ = 0 & x \in \partial O \\ > 0 & x \notin O \end{cases} \tag{1}$$

Note that points on the boundary ∂O are also part of the object.

Like most geometric modeling types, implicit objects can be used in a wide range of applications. For our work, medical applications are of special interest. In the recent research project PROREOP, CSG trees with superquadric leaves are used for modeling the femur bone and shaft. The models are used in the scope of a total hip replacement procedure, where we have to find an implant that will fit into a femur shaft that has already been shaped.

For this procedure we need to derive a verified enclosure of the distance between the femur shaft and the implant. While the direct distance computation between two superquadrics is a non trivial task even in a non verified computation [3] , in our case the bones are CSG models with superquadric leafs. We use hierarchical decomposition, in order to break the model down into less complex pieces. Some preliminary tests that have been carried out in the scope of the work [4], however showed the need for more sophisticated decomposition strategies for complex implicit objects. Especially the common octree technique with uniform space decomposition did not perform very well.

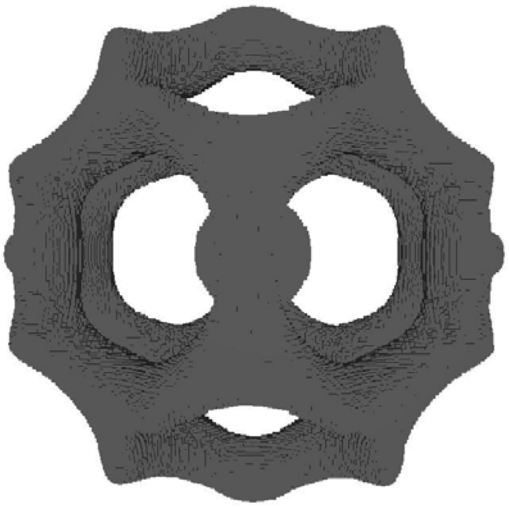


Fig. 1. The hierarchical decomposition of the implicit object given by the formula $2-\cos(x+\alpha y)+\cos(x-\alpha y)+\cos(y+\alpha z)+\cos(y-\alpha z)+\cos(z+\alpha x)+\cos(z-\alpha x)$ with $\alpha = 1.61803$ in the interval box $[-4.8, 4.8]^3$ using our novel technique with a maximum subdivision depth of 21.

In this work we discuss, how to construct a verified and tight spatial hierarchical decomposition of an object given by a complex implicit formula (cf. Fig. 1).

2 Preliminaries

2.1 Interval Arithmetic

In general a real number $x \in \mathbb{R}$ cannot be exactly represented by a floating point number as the set of all floating point numbers $\mathbb{F}$ is only a finite subset of $\mathbb{R}$. In the commonly used IEEE 754 standard the real number x is rounded to the nearest[1] floating point number x'. The same is of course true for an operation with two floating point numbers.

While often the round-off errors cancel each other out, sometimes floating point computations may produce disastrous wrong results. In medical applications, we need to make sure that our computation is really correct. Using interval arithmetic [1], a common tool for verified computations, we are a able to retrieve an interval that is guaranteed to contain the result of the exact real computation.

Let $\mathbb{I}$ denote the set of all compact intervals. An interval[2] $X = [\underline{X}, \overline{X}] \in \mathbb{I}$ consists of a lower bound $\underline{X}$ and an upper bound $\overline{X}$ with $\underline{X} \leq \overline{X}$. On a computer, the bounds are floating point numbers.

[1] In the standard rounding mode 'round to nearest even'.

[2] Intervals and interval vectors are denoted with capital letters.

We can define the basic operations between intervals. Formally, we have for $X, Y \in \mathbb{I}$ and an operation $\circ \in \{+, -, \cdot, /\}$:

$$X \circ Y = \{x \circ y | x \in X, y \in Y\}.$$

For the basic operations, simpler formulas have been derived:

$$\begin{aligned} X + Y &= [\underline{X} + \underline{Y}, \overline{X} + \overline{Y}] \\ X - Y &= [\underline{X} - \overline{Y}, \overline{X} - \underline{Y}] \\ X \times Y &= [\min(\underline{XY}, \underline{X}\overline{Y}, \overline{X}\underline{Y}, \overline{XY}), \\ &\quad\ \max(\underline{XY}, \underline{X}\overline{Y}, \overline{X}\underline{Y}, \overline{XY})] \\ X \div Y &= X \times [\tfrac{1}{\overline{Y}}, \tfrac{1}{\underline{Y}}], 0 \notin Y. \end{aligned}$$

We can implement these formulas on a computer but we have to cope with the round-off errors. This can be done using directed rounding modes[3]. We round towards $-\infty$ for calculating the lower bound and towards $+\infty$ for the upper bound.

If we replace all operations, constants and variables in a function f by their respective interval counterparts, we get the (natural) interval extension F. An interval extension has to satisfy the inclusion property:

$$F(X) \supseteq \{f(x) | x \in X\}. \tag{2}$$

As there are various ready-to-use software packages like the C-XSC library [11] available, which not only provides the basic operations but also interval extensions for elementary functions like $\sin, \cos, \exp$ etc., we assume that the interval extension of an implicit function is available.

For an interval, we will denote the width of the interval with $\mathrm{w}(X) = \overline{X} - \underline{X}$ and the midpoint with $\mathrm{mid}(X) = (\overline{X} + \underline{X})0.5$. An interval vector (box) $X \in \mathbb{I}^n$ is an axis-aligned box in n-space. Figure 2 demonstrates the two-dimensional case. The width of an interval vector is defined as the maximum over the width of its components.

Not only can interval arithmetic handle round-off errors, but it is also a very powerful tool for range analysis of functions. According to the inclusion property (2), we can calculate an enclosure of the function's true range over an interval using its interval extension.

2.2 Hierarchical Decomposition and Interval Arithmetic

The ability to calculate an enclosure of a function's range over a box is a key feature of the application of IA within the scope of spatial hierarchical decomposition. We can adapt an octree [13] to an interval octree with two simple steps:

[3] Directed rounding modes are required by the IEEE 754 standard.

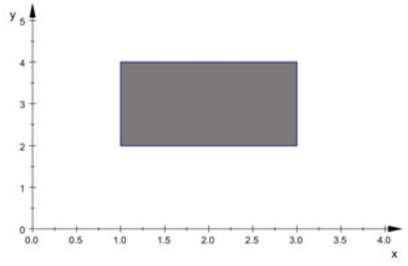


Fig. 2. The two dimensional interval vector $([1,3],[2,4])$ represents an axis-aligned box in $\mathbb{R}^2$

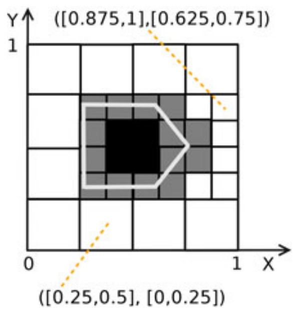


Fig. 3. An interval quadtree decomposition. The nodes are boxes.

1. Represent the octree's nodes with boxes
2. Use the implicit function's interval extension to determine the node colors

Given an object O with the implicit function f and its respective interval extension F, we can determine the color of the box X as follows:

$$\text{COLOR}(X) = \begin{cases} \text{black if } \overline{F(X)} \leq 0 \\ \text{white if } \underline{F(X)} > 0 \\ \text{gray} \quad \text{else} \end{cases}$$

If the node is black or white, we basically have a computational proof that the node lies completely inside O or is disjoint with O. However, if the node is gray, we cannot make any statement until we subdivide the node further. As it is not possible to represent arbitrary objects exactly using axis-aligned boxes, we have to define some maximum depth where the subdivision process stops. So, in general, we also have gray leaf nodes. These are uncertain areas, we do not know whether they contain a part of the object or not.

As mentioned in Sect. 1, we want to use the decomposition for distance computation as described in [4], [5]. The gray leaf nodes have direct impact on the tightness of the distance computation. We can see this in Fig. 4. Basically, we have to treat the gray leaf nodes as black for computing the lower bound and as white when computing the upper bound on the actual distance between the trees. To get a tight enclosure, it is necessary to shrink the gray leaf node's area as much as possible.

The gray leaf nodes are not only produced by the approximation of an object by axis-aligned boxes. Another source is the overestimation of IA. If we consider the inclusion property (2) more closely, we see that IA only guarantees the computation of a superset of the true range. This so-called overestimation can also lead to (unnecessary) gray leaf nodes.

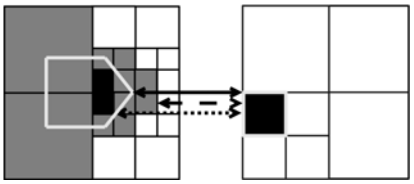


Fig. 4. Distance computation between two quadtrees. The acutal distance (continuous line) is enclosed by the lower bound (dashed line) and upper bound (dotted line).

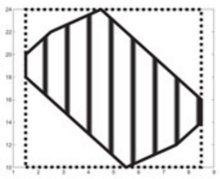


Fig. 5. The joint range of two affine forms with partial linear dependencies is a centrally symmetric polytope (shaded) and often tighter than the interval enclosure (dotted box)

2.3 Affine Arithmetic

Several more sophisticated arithmetics have been proposed to reduce the overestimation in classic IA. One is affine arithmetic [7], which is an arithmetic for verified numerics with first-order dependency tracking. In affine arithmetic, a partially unknown quantity X is represented as an affine form $\hat{X}$, which is an affine combination of a central value x_0 and error terms:

$$\hat{X} = x_0 + x_1\epsilon_1 + x_2\epsilon_2 + \cdots + x_n\epsilon_n.$$

Every error term consists of a symbolic noise variable ϵ_i and a partial deviation x_i. It is assumed that every ϵ_i lies somewhere in the interval $[-1, 1]$. There is partial linear dependency between two affine forms if they share one or more symbolic noise variables. The joint range of affine forms is a centrally symmetric polytope and not an axis-aligned box, as we can see in Fig. 5.

Affine operations can be performed straightforwardly. Given two affine forms $\hat{X}$ and $\hat{Y}$ we get:

$$\begin{aligned}
\hat{X} + \hat{X} &= (x_0 + y_0) + (x_1 + y_1)\epsilon_1 + \cdots + (x_n + y_n)\epsilon_n, \\
\alpha\hat{X} &= (\alpha x_0) + (\alpha x_1)\epsilon_1 + \cdots x_n)\epsilon_n, \\
\alpha + \hat{X} &= (x_0 + \alpha) + x_1\epsilon_1 + \cdots + x_n\epsilon_n.
\end{aligned}$$

If we perform these operations in a floating-point number system, they are unfortunately not exact. So we have to cope with the round-off errors by adding a new error term with an independent, unused noise symbol ϵ_{n+1} to enclose the error. A non-affine operation or function $f(a)$ cannot be performed straightforwardly. The basic idea is to split $f(a)$ into an affine part $f_a(a)$ approximating $f(a)$ and a non-affine part enclosed by a new error term with a previously unused noise symbol ϵ_{n+1}.

2.4 Implicit Linear Interval Estimations

Implicit linear interval estimation (ILIE) [2] is a linearization technique developed by Katja Bühler. Let O be an implicit object described by $f(x) = 0$ with $x \in \mathbb{R}^d$, its level set $\mathcal{F} = \{x|f(x) = 0\}$ and

$$L(x) := \sum_{i=1}^{d} a_i x_i + J \text{ with } J \in \mathbb{I}, x_i \in \mathbb{R} \tag{3}$$

The hyperplane segment $\mathcal{L} = \{x \in X | 0 \in L(x)\}$ is called an ILIE of O over X, if and only if

$$0 \in L(x)$$

holds for all $x \in \mathcal{F} \cap X$.

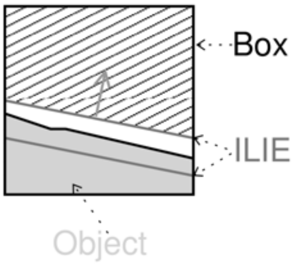


Fig. 6. An ILIE enclosing the objects boundary. Every part of the boundary is enclosed; however, the enclosure can suffer from overestimation.

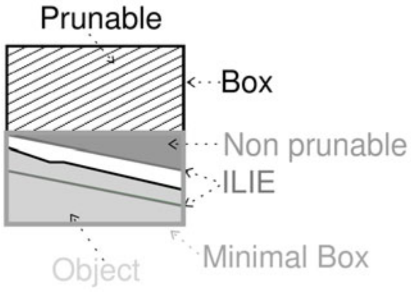


Fig. 7. Minimal outer box enclosure of the solution set after applying hull consistency. Some areas cannot be pruned even if they are not covered by the ILIE.

Informally we can say, that an ILIE is a *thick* hyperplane enclosing the boundary of an implicit object over a box X in a verified manner. An illustration is given in Fig. 6. The oriented hyperplane encloses the object boundary in the box, so it is only valid for X. It is important to note that per definition we have

$$x \in \partial O \Rightarrow 0 \in L(x) \tag{4}$$

$\forall x \in X$. However the opposite direction is not true. So

$$x \in \partial O \Leftarrow 0 \in L(x)$$

does *not* hold in general. The thickness is determined by the width of the interval parameter J in (3). It is a measure for the uncertainty and the linearization error. In most cases, $\mathrm{w}(J)$ decreases as we shrink the box X. So we get normally a better fit if we subdivide a box and recalculate the ILIE for the new smaller boxes.

The ILIE is calculated using affine arithmetic. As the paper [2] describes the process in detail, we will only give a short outline here. The process exploits the linear dependency tracking of affine arithmetic. Basically, the linear dependency of the function value $f(x)$ on the input variables x is measured using affine arithmetic. This information is used for computing the normal vector. Moreover, the central value and the linear dependencies determine the distance to the origin. Finally, the non-linear approximation errors and round-off errors are enclosed in an error interval and combined with the distance to the origin, forming the interval quantity J.

3 LIETree

In this section we propose the LIETree: A new data structure that prunes the uncertain area while still maintaining a proper hierarchy[4] and covering the whole subdivision domain. We use a binary tree as the basis of our structure. The space decomposition is non-uniform, and the nodes are interval vectors. Instead of classic IA or affine arithmetic, we use ILIE for object enclosure. But we also exploit the linearization to prune the area of uncertainty.

3.1 Pruning the Uncertainty

The pruning is based on a consistency method. In our case, we will use hull consistency (HC) as described in [10]. Given $X, Y \in \mathbb{I}$ and an equation $f(x, y) = 0$ with $x \in X, y \in Y$, we say that $x \in X$ is consistent relative to f if there exists a $y \in Y$ with $f(x, y) = 0$. It is clear that an inconsistent value x cannot fulfill the equation. Therefore, we can discard all inconsistent values without losing a solution. This idea is obviously applicable for an arbitrary number of variables. In the HC procedure, we apply this idea by solving the equation for one occurrence of one variable and replacing all other occurrences and variables with their respective interval bounds. This is repeated for all variables and will hopefully lead to sharper bounds.

Let $L(x)$ be an ILIE for the object O over $X \in \mathbb{R}^d$ then every point $x \in (X \cap O)$[5] has to satisfy

$$L(x) \leq 0$$

This follows directly from the ILIE's definition (4) and its orientation. We assume that the normal vector of L points outwards. The orientation depends on the definition of the implicit object (1). Furthermore, we can rewrite the inequality as

$$L(x) \in [-\infty, 0]$$

and deduce that every solution point x_i^* of the i-th component of X has to satisfy

$$x_i^* \in \left([-\infty, -\underline{J}] - \sum_{\substack{j=1 \\ j \neq i}}^{d} n_j X_j \right) \frac{1}{n_i}.$$

[4] 0 or 2 children.

[5] x is part of O and lies inside the ILIE's domain.

Using this information we can compute a new enclosure for X_i introducing

$$X_i' = \left(\left([-\infty, -\underline{J}] - \sum_{\substack{j=1 \\ j \neq i}}^{d} n_j X_j \right) \frac{1}{n_i} \right) \cap X_i. \tag{5}$$

We intersect the new enclosure with the original interval X_i to prevent growth of the component. So after applying (5)

$$X' \subseteq X \tag{6}$$

always holds. As the new bounds X' are always in box form and an outer approximation of the solution set[6], the bounds will still suffer from overestimation in general. That is, we have areas which are outside the ILIE but are not deletable by the HC procedure (Fig. 7)

3.2 Inversion Nodes

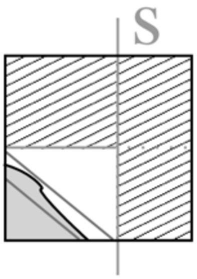


Fig. 8. The area that is not part of the object after pruning cannot be covered by a single box

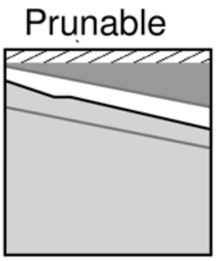


Fig. 9. Applying the HC procedure in this example cannot result in good progress, as the prunable area is too small

After applying (5) to every component for computing new bounds X', we have proven that the points $X \setminus X'$ are not part of our implicit object O. We can use this information to create a spatial hierarchical decomposition of the object. Remember that every node has an associated area described by a box. In the simplest case, we would just split X into two nodes, covering X' and $X \setminus X'$ respectively.

Unfortunately, in practice this does not work. The major problem is visualized in Fig. 8. The set $X \setminus X'$ is not necessarily a box, so it cannot be represented by a single white node. An obvious solution would be to split it at the line S. However, in this case the node including X' is enlarged and covers an area we

[6] That is, we do not lose any solution during the process.

have proven to contain no part of O. We solved this problem by introducing an inversion node. This new node type has the color white_inv and can cover the set $X \setminus X'$:

Definition 1 (Inversion node). *An inversion node has the color white_inv and occupies the area of its parent's box* $P.X$ *which is not covered by its own box* X*:* $P.X \setminus X$*.*

An inversion node can be converted exactly to a set of maximum $2d$ white nodes using Alg. 1.

Algorithm 1. Converts an inversion node to a set of white nodes

```
Data: Parent Box $P$, Pruned Box $X$, Dimension $d$
Result: Set of boxes $\mathcal{K}$ covering $P \setminus X$
$\mathcal{K} = \emptyset$;
$i = 0$;
for $i < d$ do
    if $\underline{P_i} \leq \underline{X_i}$ then
        $N = P$;
        $N_i = [\underline{P_i}, \underline{X_i}]$;
        $\mathcal{K} = \mathcal{K} \cup N$;
    end
    if $\overline{P_i} > \overline{X_i}$ then
        $N = P$;
        $N_i = [\overline{X_i}, \overline{P_i}]$;
        $\mathcal{K} = \mathcal{K} \cup N$;
    end
    $P_i = [\underline{X_i}, \overline{X_i}]$;
    $i = i + 1$;
end
return $\mathcal{K}$;
```

3.3 Insufficient Progress

The pruning process is often plagued by bad progress (Fig. 9), especially when the box X is relatively large and linearization is bad. In this example, the maximum prunable area is very small. We add white nodes for the pruned area, only if the progress is *large enough*. If it is not, we use a normal subdivision procedure that is, we bisect the box using the midpoint of the longest axis of X.

But when is the progress *large enough*? A similar problem occurs in interval global optimization, where we need to judge whether an interval contractor leads to sufficient progress. Hansen and Walster describe in [10] a heuristic method, which can be adapted for our algorithm. Let X be the original box and X' the pruned box. Then, we define

$$D = 0.25\mathrm{w}(X) - \max_{1 \le i \le d} (\mathrm{w}(X_i) - \mathrm{w}(X_i')) \qquad (7)$$

Note that the maximum term is never negative as (6) always holds. The progress is sufficient if $D \le 0$.

3.4 Construction Algorithm

Formally a LIETree node in our structure is a 6-tuple $\mathcal{N} = (X, I, F, \mathcal{C}, T, P)$. Here X is the interval vector containing the covered area[7], I contains the ILIE for this node, F a pointer to its parent, $\mathcal{C}$ is a set containing pointers to its children, T is the node color and P is the pruned area.

The pruned area results from the pruning processes that yield insufficient progress. In such a case, the pruned area is propagated to the child nodes in the regular bisection progress. As $X \setminus P$ does not belong to the object, and $P \subseteq X$ always holds we can just work on P instead of X. In general, this results in tighter codomain enclosures of the implicit function.

The complete algorithm for splitting a LIETree node is given in Alg. 2. First, we try to recolor our current node either black or white (2, 3). In some cases it may be necessary to create an inversion node (4). If this is not possible, we try to prune the uncertain area (5) and recolor the node white if the whole area has been deleted during this step (6). Next, we measure the progress (7). If the progress is sufficient (8), we create an inversion node for the pruned area and a gray one for the rest. Otherwise, we just use the common bisection procedure (9).

4 Test Results

We have compared our new structure with common uniform subdivision schemes, using binary trees with interval arithmetic (BinTree IA) and affine arithmetic (BinTree AA) for enclosure. Our seven test surfaces are listed in Tab. 1. The test criteria are the uncertainty (area of gray leaf nodes), the total number of nodes and the computation time. As subdivision depths, we have chosen 10, 15, 20 and 25. The test system was a Intel Core i7-860 system with 8 GB of memory running under Ubuntu Linux 10.04. For interval arithmetic, we used the C-XSC library and, for affine arithmetic, the C++ Affine Arithmetic Library libaffa [9].

It is a well known phenomena that IA often performs better for relatively large boxes than more sophisticated arithmetics. This also occurred during our tests, where the classic IA evaluation performed better than the LIETree or BinTree AA for low subdivision depths. To get a better enclosure at the expense of computational effort, we can perform an intersection with the natural interval extension's result. To do so, we replace Line 1 in Alg. 2 by

$$E = I(P) \cap F(P)$$

where F is the natural interval extension of the implicit function. This is called *LIETree (intersect.)* in our comparison.

[7] Except for white_inv nodes (Subsect. 3.2).

Algorithm 2. Splits a LIETree node

```
Data: Current Node $\mathcal{N} = (X, I, F, \mathcal{C}, T, P)$
Result: Set of newly created nodes $\mathcal{S}$
.
$\mathcal{S} = \emptyset$;
1  $E = I(P)$;
2  if $\underline{E} > 0$ then
       T = white;
   else if $\overline{E} \leq 0$ then
3      if $X == P$ then
           T = black;
4      else
           $\mathcal{S} = \mathcal{S} \cup (P, \textbf{null}, \mathcal{N}, \emptyset, \text{white\_inv}, P)$;
           $\mathcal{S} = \mathcal{S} \cup (P, \textbf{null}, \mathcal{N}, \emptyset, \text{black}, P)$;
       end
   else
5      Calculate $X'$ using (5);
6      if $X' == \emptyset$ then
           T = white
       else
7          Calculate $D$ using (7);
8          if $D \leq 0$ then
               $\mathcal{S} = \mathcal{S} \cup (X', \textbf{null}, \mathcal{N}, \emptyset, \text{white\_inv}, X')$;
               $\mathcal{S} = \mathcal{S} \cup (X', \text{calc\_ilie}(X'), \mathcal{N}, \emptyset, \text{gray}, X')$;
9          else
               Determine longest axis $i$ of $X'$;
               Bisect $X$ using axis $i$ and mid($X'[i]$) into $X^{(0)}, X^{(1)}$;
               Bisect $X'$ using axis $i$ and mid($X'[i]$) into $X'^{(0)}, X'^{(1)}$;
               $\mathcal{S} = \mathcal{S} \cup (X^{(0)}, \text{calc\_ilie}(X'^{(0)}), \mathcal{N}, \emptyset, \text{gray}, X'^{(0)})$;
               $\mathcal{S} = \mathcal{S} \cup (X^{(1)}, \text{calc\_ilie}(X'^{(1)}), \mathcal{N}, \emptyset, \text{gray}, X'^{(1)})$;
           end
       end
   end
   return $\mathcal{S}$;
```

The averaged uncertainty is given in Fig. 10. While, for the low subdivision depth of 10, the BinIA tree and the LIETree (intersect.) structure perform best, for medium and high subdivision depths the LIETree performs considerably better than the BinTree IA and the BinTree AA, and nearly as well as the LIETree (intersect.). If we look at the computation times summarized in Fig. 11, we see that computing the LIETree (intersect.) is very time-consuming. The LIETree however, has a very low overhead with respect to time compared to BinTree IA and especially BinTree AA. Fig. 12 shows that the new structure uses significantly fewer nodes.

Table 1. Implicit surfaces for our test series

Bretzel 5	$(x^2 + y^2/4 - 1)^2 \cdot (x^2/4 + y^2 - 1) + z^2 - 0.5$
Sphere Hole	$(y - x^2 - y^2 + 1)^4 + (x^2 + y^2 + z^2)^4 - 1$
Dodecahedral	$2 - \cos(x + \alpha y) + \cos(x - \alpha y) + \cos(y + \alpha z) + \cos(y - \alpha z) + \cos(z + \alpha x) + \cos(z - \alpha x)$ with $\alpha = 1.61803$
Heart	$(x^2 + 2.25y^2 + z^2 - 1)^3 - x^2z^3 - 0.1125y^2z^3$
Cube	$x^6 + y^6 + z^6 - 1$
Klein's bottle	$(x^2 + y^2 + z^2 + 2y - 1)((x^2 + y^2 + z^2 - 2y - 1)^2 - 8z^2) + 16xz(x^2 + y^2 + z^12 - 2y - 1)$
Sphere	$x^2 + y^2 + z^2 - 1$

5 Conclusion and Future Work

5.1 Summary

In this paper we presented a novel technique for creating a verified spatial decomposition of implicit objects. Featuring non-uniform box-based space decomposition and the use of implicit linear interval estimations, the new technique results in tighter object fitting and reduces the area of uncertainty compared to common techniques based on interval arithmetic or affine arithmetic. In combination with the new inversion node, the tighter fitting reduces the number of nodes significantly. However, the computation times are slightly increased, as the construction process is more complicated. Also, our structure suffers from large overestimation at low subdivision depths. This is a problem common to most techniques utilizing more sophisticated arithmetics like affine arithmetic. Using the intersection with standard interval arithmetic, it can be solved, but at the price of increased computation time.

5.2 Application to CSG Models and Superquadrics

We can conclude that our goal of a tighter fitting has been reached for the surfaces we tested. However, no test has been done with superquadrics because it is difficult to evaluate them with affine arithmetic. A superquadric F can be described by the implicit formula

$$F(x, y, z) = \left(\left(\frac{x}{a_1} \right)^{\frac{2}{\epsilon_2}} + \left(\frac{y}{a_2} \right)^{\frac{2}{\epsilon_2}} \right)^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{z}{a_3} \right)^{\frac{2}{\epsilon_1}} - 1 \tag{8}$$

where the exponents ϵ_1 and ϵ_2 are positive real numbers. Therefore the arguments of $(\cdot)^{\overline{\frac{1}{\epsilon_{\{1,2\}}}}}$ have to be positive. This holds for superquadrics because the arguments are squared. But the square of an affine form is not always completely positive due to overestimation, so we cannot evaluate these powers. Here, the use of an affine quadratic form [12] could solve the problem.

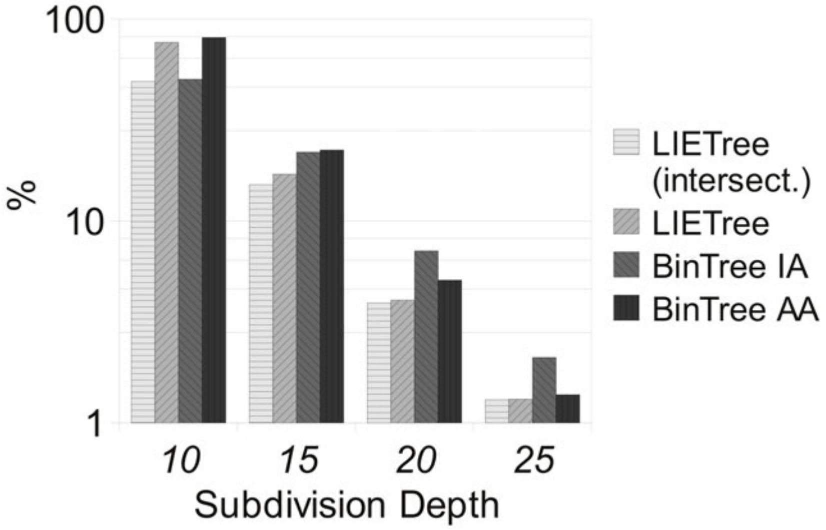
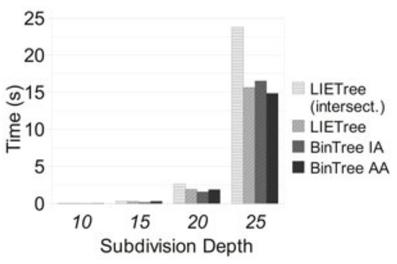


Fig. 10. Averaged uncertainty for all test cases on a logarithmic scale



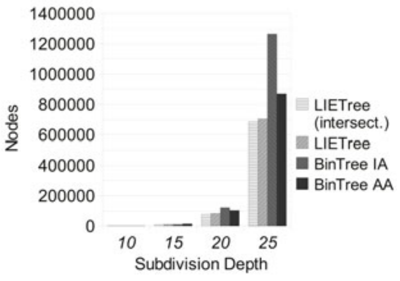


Fig. 11. Averaged computation times

Fig. 12. Averaged number of nodes

Also, the application to CSG models remains undetermined. For this, we need a way to propagate the linear dependency information through set theoretic operations. One option for this may be R-functions [14], as already used in [8]. Another option might be a hybrid approach in which we use common uniform space subdivision and combine it with the CSG simplification algorithm from [6]. We then can apply our LIETree structure as soon as the simplification procedure produces a single CSG leaf node for a box.

References

1. Alefeld, G., Herzberger, J.: Introduction to interval computations. Academic Press, New York (1983)
2. Bühler, K.: Implicit linear interval estimations. In: Proceedings of the 18th Spring Conference on Computer Graphics, p. 132. ACM, New York (2002)

3. Chakraborty, N., Peng, J., Akella, S., Mitchell, J.E.: Proximity queries between convex objects: An interior point approach for implicit surfaces. IEEE Transactions on Robotics 24(1), 211–220 (2008)
4. Cuypers, R., Kiel, S., Luther, W.: Automatic femur decomposition, reconstruction and refinement using superquadric shapes. In: Proceedings of the IASTED International Conference, vol. 663, p. 59 (2009)
5. Dyllong, E., Grimm, C.: A modified reliable distance algorithm for octree-encoded Ojects. PAMM 7(1), 4010015–4010016 (2007)
6. Dyllong, E., Grimm, C.: Verified adaptive octree representations of constructive solid geometry objects. In: Simulation und Visualisierung, pp. 223–235. Citeseer (2007)
7. de Figueiredo, L., Stolfi, J.: Self-validated numerical methods and applications. IMPA, Rio de Janeiro (1997)
8. Fryazinov, O., Pasko, A., Comninos, P.: Fast reliable interrogation of procedurally defined implicit surfaces using extended revised affine arithmetic. Computers & Graphics (2010) (in Press) (Corrected Proof), http://www.sciencedirect.com/science/article/B6TYG-50KC6R9-1/2/41965886a337f58408f2fdf8b1d2d4bc
9. Gay, O., Coeurjolly, D., Hurst, N.J.: libaffa webpage. online
10. Hansen, E., Walster, G.W.: Global optimization using interval analysis. Marcel Dekker, New York (2004)
11. Klatte, R.: C-XSC: A C++ class library for extended scientific computing. Springer-Verlag New York, Inc., Secaucus (1993); translator-Corliss, G. F
12. Messine, F., Touhami, A.: A general reliable quadratic form: An extension of affine arithmetic. Reliable Computing 12, 171–192 (2006), http://dx.doi.org/10.1007/s11155-006-7217-4, doi:10.1007/s11155-006-7217-4
13. Samet, H.: Foundations of multidimensional and metric data structures. Morgan Kaufmann, San Francisco (2006)
14. Shapiro, V.: Theory of R-functions and applications: A primer (1991)

Image Separation Using Wavelets and Shearlets

Gitta Kutyniok and Wang-Q Lim

Institute of Mathematics, University of Osnabrück,
49069 Osnabrück, Germany
{kutyniok,wlim}@uni-osnabrueck.de

Abstract. In this paper, we present an image separation method for separating images into point- and curvelike parts by employing a combined dictionary consisting of wavelets and compactly supported shearlets utilizing the fact that they sparsely represent point and curvilinear singularities, respectively. Our methodology is based on the very recently introduced mathematical theory of geometric separation, which shows that highly precise separation of the morphologically distinct features of points and curves can be achieved by ℓ^1 minimization. Finally, we present some experimental results showing the effectiveness of our algorithm, in particular, the ability to accurately separate points from curves even if the curvature is relatively large due to the excellent localization property of compactly supported shearlets.

Keywords: Geometric separation, ℓ_1 minimization, sparse approximation, shearlets, wavelets.

1 Introduction

The task of separating an image into its morphologically different contents has recently drawn a lot of attention in the research community due to its significance for applications. In neurobiological imaging, it would, for instance, be desirable to separate 'spines' (pointlike objects) from 'dendrites' (curvelike objects) in order to analyze them independently aiming to detect characteristics of Alzheimer disease. Also, in astronomical imaging, astronomers would often like to separate stars from filaments for further analysis, hence again separating point- from curvelike structures. Successful methodologies for efficiently and accurately solving this task can in fact be applied to a much broader range of areas in science and technology including medical imaging, surveillance, and speech processing.

Although the problem of separating morphologically distinct features seems to be intractable – the problem is underdetermined, since there is only one known data (the image) and two or more unknowns – there has been extensive studies on this topic. The book by Meyer [18] initiated the area of image decomposition, in particular, the utilization of variational methods. Some years later, Starck, Elad, and Donoho suggested a different approach in [19] coined 'Morphological Component Analysis', which proclaims that such a separation task might be possible provided that we have prior information about the type of features to be extracted and provided that the morphological difference between those is strong enough. For the separation of point- and curvelike features, it was in fact recently even theoretically proven in [3] that ℓ_1 minimization solves this task with arbitrarily high precision exploring a combined dictionary of wavelets and curvelets. Wavelets provide optimally sparse expansions for pointlike structures, and curvelets provide optimally sparse expansions for curvelike structures. Thus ℓ_1 minimization applied to the expansion coefficients of the original image into this combined dictionary forces the pointlike structures into the wavelet part and the curvelike structures into the curvelet part, thereby automatically separating the image. An associated algorithmic approach using wavelets and curvelets has been implemented in MCALab[1].

Recently, a novel directional representation system – so-called shearlets – has emerged which provides a unified treatment of continuum models as well as digital models, allowing, for instance, a precise resolution of wavefront sets, optimally sparse representations of cartoon-like images, and associated fast decomposition algorithms; see the survey paper [11]. Shearlet systems are systems generated by one single generator with parabolic scaling, shearing, and translation operators applied to it, in the same way wavelet systems are dyadic scalings and translations of a single function, but including a directionality characteristic owing to the additional shearing operation (and the anisotropic scaling). The shearing operation in fact provides a more favorable treatment of directions, thereby ensuring a unified treatment of the continuum and digital realm as opposed to curvelets which are rotation-based in the continuum realm, see [1].

Thus, it is natural to ask whether also a combined dictionary of wavelets and shearlets might be utilizable for separating point- and curvelike features, the advantage presumably being a faster scheme, a more precise separation, and a direct applicability of theoretical results achieved for the continuum domain. And, in fact, the theoretical results from [3] based on a model situation were shown to also hold for a combined dictionary of wavelets and shearlets [2]. Moreover, numerical results give evidence to the superior behavior of shearlet-based decomposition algorithms when compared to curvelet-based algorithms; see [11] for a comparison of ShearLab[2] with CurveLab[3].

In this paper, we will present a novel approach to the separation of point- and curvelike features exploiting a combined dictionary of wavelets and shearlets as well as utilizing block relaxation in a particular way. Numerical results give evidence that indeed the previously anticipated advantages hold true, i.e., that this approach is superior to separation algorithms using wavelets and curvelets such as MCALab in various ways, in particular, our algorithm is faster and provides a more precise separation, in particular, if the curvature of the curvilinear part is large. In the spirit of reproducible research [5], our algorithm is included in the freely available ShearLab toolbox.

[1] MCALab (Version 120) is available from `http://jstarck.free.fr/jstarck/Home.html`

[2] ShearLab (Version 1.0) is available from `http://www.shearlab.org`

[3] CurveLab (Version 2.1.2) is available from `http://www.curvelet.org`

This paper is organized as follows. In Section 2, we introduce the multiscale system of shearlets, and Section 3 reviews the mathematical theory of geometric separation of point- and curvelike features. Our novel algorithmic approach is presented in Section 4 with numerical results discussed in Section 5.

2 Shearlets

In most multivariate problems, important features of the considered data are concentrated on lower dimensional manifolds. For example, in image processing an edge is an 1D curve that follows a path of rapid change in image intensity. Recently, the novel directional representation system of shearlets [14,8] has emerged to provide efficient tools for analyzing the intrinsic geometrical features of a signal using anisotropic and directional window functions. In this approach, directionality is achieved by applying integer powers of a shear matrix, and those operations preserve the structure of the integer lattice which is crucial for digital implementations. In fact, this key idea leads to a unified treatment of the continuum as well as digital realm, while still providing optimally sparse approximations of anisotropic features. As already mentioned before, shearlet systems are generated by parabolic scaling, shearing, and translation operators applied to one single generator. Let us now be more precise and formally introduce shearlet systems in 2D.

We first start with some definitions for later use. For $j \geq 0$ and $k \in \mathbb{Z}$, let

$$A_{2^j} = \begin{pmatrix} 2^j & 0 \\ 0 & 2^{j/2} \end{pmatrix}, \quad \tilde{A}_{2^j} = \begin{pmatrix} 2^{j/2} & 0 \\ 0 & 2^j \end{pmatrix}, \quad \text{and} \quad S_k = \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}.$$

We can now define so-called cone-adapted discrete shearlet systems, where the term 'cone-adapted' originates from the fact that these systems tile the frequency domain in a cone-like fashion. For this, let c be a positive constant, which will later control the sampling density. For $\phi, \psi, \tilde{\psi} \in L^2(\mathbb{R}^2)$, the *cone-adapted discrete shearlet system* $SH(\phi, \psi, \tilde{\psi}; c)$ is then defined by

$$SH(\phi, \psi, \tilde{\psi}; c) = \Phi(\phi; c) \cup \Psi(\psi; c) \cup \tilde{\Psi}(\tilde{\psi}; c),$$

where

$$\begin{aligned}
\Phi(\phi; c) &= \{\phi(\cdot - cm) : m \in \mathbb{Z}^2\}, \\
\Psi(\psi; c) &= \{\psi_{j,k,m} = 2^{\frac{3}{4}j} \psi(S_k A_{2^j} \cdot - cm) : j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, m \in \mathbb{Z}^2\}, \\
\tilde{\Psi}(\tilde{\psi}; c) &= \{\tilde{\psi}_{j,k,m} = 2^{\frac{3}{4}j} \tilde{\psi}(S_k^T \tilde{A}_{2^j} \cdot - cm) : j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, m \in \mathbb{Z}^2\}.
\end{aligned}$$

In [9], a comprehensive theory of compactly supported shearlet frames is provided, i.e., systems with excellent spatial localization. It should also be mentioned that in [12] a large class of compactly supported shearlet frames were shown to provide optimally sparse approximations of images governed by curvilinear structures, in particular, so-called *cartoon-like images* as defined in [1]. This fact will be explored in the sequel.

3 Mathematical Theory of Geometric Separation

In [19], a novel image separation method – Morphological Component Analysis (MCA) – based on sparse representations of images was introduced. In this approach, it is assumed that each image is the linear combination of several components that are morphologically distinct – for instance, points, curves, and textures. The success of this method relies on the assumption that each of the components is sparsely represented in a specific representation system. The key idea is then the following: Provided that such representation systems are identified, the usage of a pursuit algorithm searching for the sparsest representation of the image with respect to the dictionary combining all those specific representation systems will lead to the desired separation.

Various experimental results in [19] show the effectiveness of this method for image separation however without any accompanying mathematical justification. Recently, the first author of this paper and Donoho developed a mathematical framework in [3] within which the notion of successful separation can be made definitionally precise and can be mathematically proven in case of separating point- from curvelike features, which they coined *Geometric Separation.* One key ingredient of their analysis is the consideration of clustered sparsity properties measured by so-called *cluster coherence.* In this section, we briefly review this theoretical approach to the Geometric Separation Problem, which will serve as the foundation for our algorithm.

3.1 Model Situation

As a mathematical model for a composition of point- and curvelike structures, we consider the following two components: As a 'point-like' object, we consider the function $\mathcal{P}$ which is smooth except for point singularities and is defined by

$$\mathcal{P} = \sum_{i=1}^{P} |x - x_i|^{-3/2}.$$

As a 'curve-like' object, we consider the distribution $\mathcal{C}$ with singularity along a closed curve $\tau : [0,1] \to \mathbb{R}^2$ defined by

$$\mathcal{C} = \int \delta_{\tau(t)} dt.$$

Then our model situation is the sum of both, i.e.,

$$f = \mathcal{P} + \mathcal{C}. \qquad (1)$$

The Geometric Separation Problem now consists of recovering $\mathcal{P}$ and $\mathcal{C}$ from the observed signal f.

3.2 Chosen Dictionary

As we indicated before, it is now crucial to choose two representations systems each of which sparsely represents one of the morphologically different components in the Geometric Separation Problem. Our sparse approximation result, described in the previous section, suggests that curvilinear singularities can be sparsely represented by shearlets. On the other hand, it is well known that wavelets can provide optimally sparse approximations of functions which are smooth apart from point singularities. Hence, we choose the overcomplete system within which we will expand the signal f as a composition of the following two systems:

– *Orthonormal Separable Meyer Wavelets*: Band-limited wavelets which form an orthonormal basis of isotropic generating elements.
– *Bandlimited Shearlets*: A directional and anisotropic tight frame generated by a band-limited shearlet generator ψ defined in Section 2.

3.3 Subband Filtering

Since the scaling subbands of shearlets and wavelets are similar we can define a family of filters $(F_j)_j$ which allows to decompose a function f into pieces f_j with different scales j depending on those subbands. The piece f_j associated to subband j arises from filtering f using F_j by

$$f_j = F_j * f,$$

resulting in a function whose Fourier transform $\hat{f}_j$ is supported on the scaling subband of scale j of the wavelet as well as the shearlet frame. The filters are defined in such way, that the original function can be reconstructed from the sequence $(f_j)_j$ using

$$f = \sum_j F_j * f_j, \quad f \in L^2(\mathbb{R}^2).$$

We can now exploit these tools to attack the Geometric Separation Problem scale-by-scale. For this, we filter the model problem (1) to derive the sequence of filtered images

$$f_j = \mathcal{P}_j + \mathcal{C}_j \quad \text{for all scales } j.$$

3.4 ℓ_1 Minimization Problem

Let now Φ_1 and Φ_2 be an orthonormal basis of band-limited wavelets and a tight frame of band-limited shearlets, respectively. Then, for each scale j, we consider the following optimization problem:

$$(\hat{W}_j, \hat{S}_j) = \text{argmin}_{W_j, S_j} \|\Phi_1^T W_j\|_1 + \|\Phi_2^T S_j\|_1 \quad \text{subject to} \quad f_j = W_j + S_j. \quad (2)$$

Notice that $\Phi_1^T W_j$ and $\Phi_2^T S_j$ are the wavelet and shearlet coefficients of the signals W_j and S_j, respectively. Notice that our objective is *not* on searching for the sparsest expansion in a wavelet-shearlet dictionary, but on separation. Thus we can avoid an extensive, presumably numerically instable search by minimizing specific coefficients, namely the *analysis* in contrast to the *synthesis* coefficients, for each possible separation $f_j = W_j + S_j$.

We wish to further remark, that here the ℓ_1 norm is placed on the *analysis* rather than the *synthesis* coefficients to avoid numerical instabilities due to the redundancy of the shearlet frame.

3.5 Theoretical Result

The theoretical result of the precision of separation of f_j via (2) proved in [3] and [2] can now be stated in the following way:

Theorem 1 ([3] and [2]). *Let $\hat{W}_j$ and $\hat{S}_j$ be solutions to the optimization problem (2) for each scale j. Then we have*

$$\frac{\|\mathcal{P}_j - \hat{W}_j\|_2 + \|\mathcal{C}_j - \hat{S}_j\|_2}{\|\mathcal{P}_j\|_2 + \|\mathcal{C}_j\|_2} \to 0, \quad j \to \infty.$$

This result shows that the components $\mathcal{P}_j$ and $\mathcal{C}_j$ are recovered with asymptotically arbitrarily high precision at very fine scales. The energy in the pointlike component is completely captured by the wavelet coefficients, and the curvelike component is completely contained in the shearlet coefficients. Thus, the theory evidences that the Geometric Separation Problem can be satisfactorily solved by using a combined dictionary of wavelets and shearlets and an appropriate ℓ_1 minimization problem.

3.6 Extensions

Our numerical scheme for image separation, which we will present in detail in the next section, will use a shift invariant wavelet tight frame and a compactly supported shearlet frame as opposed to orthonormal Meyer wavelets and band-limited shearlets required for Theorem 1. Hence this deserves some comments. Firstly, Theorem 1 is based on an abstract separation estimate which holds for any pair of frames, provided certain relative sparsity and cluster coherence conditions with respect to the components of the data to be separated are satisfied (cf. [3]). Secondly, using the recently introduced concept of *sparsity equivalence* (see [2,10]), results requiring sparsity and coherence conditions can be transferred from one system (set of systems) to another by 'merely' considering particular decay conditions of the cross-Grammian matrix (matrices). We strongly believe that this framework allows a similar result as Theorem 1 for the pair of a shift invariant wavelet tight frame and a compactly supported shearlet frame. Since the focus of this paper is however on the introduction of the numerical scheme, such a highly technical, theoretical analysis is beyond the scope of this paper and will be treated in a subsequent work.

4 Our Algorithmic Approach to the Geometric Separation Problem

In this section, we present our algorithmic approach to the Geometric Separation Problem of separating point- from curvelike features by using a combined dictionary of wavelets and shearlets. The ingredients of the algorithm will be detailed below.

4.1 General Scheme

In practice, the observed signal f is often contaminated by noise which requires an adaption of the optimization problem (2). As proposed in numerous publications, one typically considers a modified optimization problem – so-called Basis Pursuit Denoising (BPDN) – which can be obtained by relaxing the constraint in (2) in order to deal with noisy observed signals (see [6]). For each scale j, the optimization problem then takes the form:

$$(\hat{W}_j, \hat{S}_j) = \text{argmin}_{W_j, S_j} \|\Phi_1^T W_j\|_1 + \|\Phi_2^T S_j\|_1 + \lambda \|f_j - W_j - S_j\|_2^2. \qquad (3)$$

In this new form, the additional content in the image – the noise – characterized by the property that it can not be represented sparsely by either one of the two representation systems will be allocated to the residual $f_j - W_j - S_j$. Hence, performing this minimization, we not only separate point- and curvelike objects, which were modeled by $\mathcal{P}_j$ and $\mathcal{C}_j$ in Subsection 3.1, but also succeed in removing an additive noise component as a by-product. Of course, solving the optimization problem (3) for all relevant scales j is computationally expensive.

4.2 Preprocessing

To avoid high complexity, we observe that the frequency distribution of point- and curvelike components is highly concentrated on high frequencies. Hence it would be essentially sufficient for achieving accurate separation to solve (3) for only sufficiently large scales j, as also evidenced by Theorem 1. This idea leads to a simplification of the problem (3) by modifying the observed signal f as follows: We first consider bandpass filters $F_0, \ldots, F_L$, where $(F_j)_{j=0,\ldots,L}$ is the family of bandpass filters defined in Subsection 3.3 up to scale L, and F_0 is a lowpass filter. Thus the observed signal f satisfies

$$f = \sum_{j=0}^{L} F_j * f_j \quad \text{with } f_j = F_j * f.$$

For each scale j, we now carefully choose a non-uniform weight $w_j > 0$ satisfying, in particular, $w_j < w_{j'}$, if $j < j'$. These weights are then utilized for a weighted reconstruction of f resulting in a newly constructed signal $\tilde{f}$ by computing

$$\tilde{f} = \sum_{j=0}^{L} w_j \cdot (F_j * f_j). \qquad (4)$$

In this way, the two morphological components, namely points and curves, can be enhanced by suppressing the low frequencies.

While emphasizing that certainly other weights can be applied in our scheme, for the numerical tests presented in this paper we chose $L = 3$, i.e., 4 subbands, and weights $w_0 = 0, w_1 = 0.1, w_2 = 0.7$, and $w_3 = 0.7$. This coincides with the intuition that a strong weight should be assigned to a band on which the power spectrum of the underlying morphological contents is highly concentrated. We do not claim that this is necessarily the optimal choice, and a comprehensive mathematical optimality analysis is beyond our reach at this moment. To our mind, the numerical results though justify this choice.

4.3 Solver for the ℓ_1 Minimization Problem

Using the reweighted reconstruction of f from (4) coined $\tilde{f}$, we now consider the following new minimization problem:

$$(\hat{W}, \hat{S}) = \mathrm{argmin}_{W,S} \|\Phi_1^T W\|_1 + \|\Phi_2^T S\|_1 + \lambda \|\tilde{f} - W - S\|_2^2. \qquad (5)$$

Note that the frequency distribution of $\tilde{f}$ is highly concentrated on the high frequencies – in other words, scaling subbands of large scales j –, and Theorem 1 justifies our expectation of a very precise separation using $\tilde{f}$ instead of f. Even more advantageous, the reduced problem (5) no longer involves different scales j, and hence can be efficiently solved by various fast numerical schemes. In our separation scheme, we use the same optimization method as the one used in MCALab to solve (5) now applied to a combined wavelet-shearlet dictionary. We refer to [7] for a detailed description of an algorithmic approach to solve (5); see also [6]. In the following subsections, we discuss the particular form of the matrices Φ_1^T and Φ_2^T which encode the wavelet and shearlet transform in the minimization problem (5) we aim to solve.

4.4 Wavelet Transform

Let us start with the wavelet transform. The undecimated digital wavelet transform is certainly the most fitting version of the wavelet transform for the filtering of data, and hence this is what we utilize also here. This transform is obtained by skipping the subsampling, thereby yielding an overcomplete transform, which in addition is shift-invariant. The redundancy factor of this transform is $3J + 1$, where J is the number of decomposition levels. We refrain from further details and merely refer the reader to [17].

4.5 Shearlet Transform

For the shearlet transform, we employ the digital shearlet transform implemented by 2D convolution with discretized compactly supported shearlets, which was introduced in [16], see also [13]. In the earlier work [15], an faithful digitalization of the continuum domain shearlet transform using compactly supported shearlets generated by separable functions has been developed. However, firstly, this algorithmic realization allows only a limited directional selectivity due to separability and, secondly, compactly supported shearlets generated by separable functions do not form a tight frame which causes an additional computational effort to approximate the inverse of the shearlet transform by iterative methods. These problems have been resolved in [16,13] by using non-separable compactly supported generators, and we now summarize this procedure.

In the sequel, we will discuss the implementation strategy for computing the shearlet coefficients $\langle f, \psi_{j,k,m}\rangle$ only for $\psi_{j,k,m} \in \Psi(\psi, 1)$. The same procedure can be applied to shearlets in $\tilde{\Psi}(\tilde{\psi}, 1)$ except for switching the role of variables. Without loss of generality, let $j/2$ be an integer. For $J > 0$ fixed, assume that $f(x) = \sum_{n\in\mathbb{Z}^2} f_J(n)2^L\phi(2^Lx - n)$, where ϕ is a 2D separable scaling function of the form $\phi_1(x_1)\phi_1(x_2)$ satisfying

$$\phi(x) = \sum_{n\in\mathbb{Z}^2} h(n)2\phi(2x - n). \tag{6}$$

Let ψ be a 2D separable wavelet defined by $\psi(x_1, x_2) = \psi_1(x_1)\phi_1(x_2)$, where ψ_1 is a 1D wavelet. Further, assume that ψ can be written as

$$\psi(x) = \sum_{n\in\mathbb{Z}^2} w(n)2\phi(2x - n), \tag{7}$$

where $w(n)$ are 2D separable wavelet filter coefficients associated with scaling filter coefficients $h(n)$. For each $j \geq 0$, define the (non-separable) shearlet generator ψ_j^{non} by

$$\hat{\psi}_j^{\text{non}}(\xi) = P_{J-j/2}(\xi)\hat{\psi}(\xi), \tag{8}$$

where $P_\ell(\xi) = P(2^{\ell+1}\xi_1, \xi_2)$ for $\ell \geq 0$ and the trigonometric polynomial P is a 2D fan filter (c.f. [4]). To implement $\langle f(\cdot), \psi_{j,k,m}^{\text{non}}(\cdot)\rangle = \langle f(\cdot), \psi_{j,0,m}^{\text{non}}(S_{2^{-j/2}k}\cdot)\rangle$, we make two observations: Firstly, the functions $\psi_{j,0,m}^{\text{non}}$ are wavelets generated by refinement equations (6), (7) and (8). Thus, for each j, there exists an associated 2D wavelet filter w_j. Secondly, the shear operator $S_{2^{-j/2}k}$ can be faithfully discretized by the digital shear operator $S^d_{2^{-j/2}k}$ (see [15,16], also [13]). The *digital (non-separable) shearlet transform* is then, using the shearlet filters $\psi^d_{j,k} = S^d_{2^{-j/2}k}(w_j)$, defined by

$$SH(f_J)(m_1, m_2) = (f_J * \psi^d_{j,k})(2^{J-j}m_1, 2^{J-j/2}m_2) \quad \text{for } f_J \in \ell^2(\mathbb{Z}^2).$$

If downsampling by A_{2^j} is omitted, a shift invariant shearlet transform $(f_J * \psi^d_{j,k})(m_1, m_2)$ is obtained, in which case dual shearlet filters $\tilde{\psi}^{\text{d}}_{j,k}$ can be easily computed by deconvolution. We then obtain the reconstruction formula

$$f_J = \sum_{j,k}(f_J * \psi^{\text{d}}_{j,k}(-\,\cdot)) * \tilde{\psi}^{\text{d}}_{j,k}.$$

5 Numerical Results

In this section, we present and discuss some numerical results of our proposed scheme for separating point- and curvelike features. In each experiment we compare our scheme, which is freely available in the ShearLab[4] toolbox, with the separation algorithm MCALab[5]. In contrast to our algorithm, MCALab uses wavelets and curvelets to separate point- and curvelike components, and we refer to [7] for more details on the algorithm.

5.1 Comparison by Visual Perception

Aiming first at comparison by visual perception, we choose an artificial image I composed of two subimages P and C, where P solely contains pointlike structures and C different curvelike structures (see Figures 1(a) and (b)). We further add white Gaussian noise to $I = P + C$, shown in Figure 1(c).

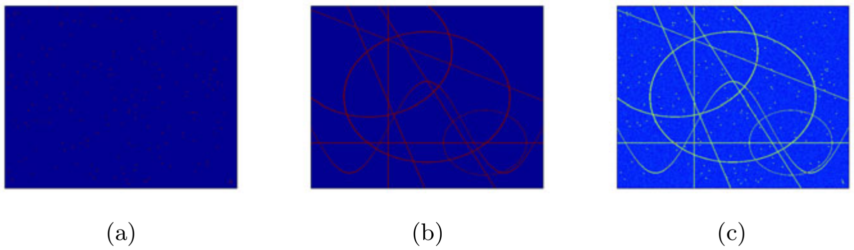


Fig. 1. (a) P: Image of points. (b) C: Image of curves. (c) Noisy image (512 × 512).

Our scheme consists of two parts: Preprocessing of the image as described in Subsection 4.2, followed by separation using a combined wavelet-shearlet dictionary as described in Subsections 4.3-4.5. First, we will focus on the preprocessing step, and apply MCALab with and without our preprocessing step – due to limited space we just mention that our scheme is similarly positively affected by preprocessing. Figures 2(a) and (b) then visually indicate that preprocessing indeed significantly improves the accuracy of separation.

To achieve a fair comparison, we now apply both schemes to the same *preprocessed* image as defined in (4). Figures 3(a)-(d) and Figures 4(a)-(b) show the comparison results. In Figure 3(c), it can be observed that the curvelet transform performs well for extracting lines due to excellent directional selectivity. However, some part of the curve is missed and appears in the pointlike part, see also Figure 4(a). This error becomes worse with growing curvature. In contrast to this, compactly supported shearlets provide much better spatial localization

[4] ShearLab (Version 1.1) is available from `http://www.shearlab.org`

[5] MCALab (Version 120) is available from `http://jstarck.free.fr/jstarck/Home.html`

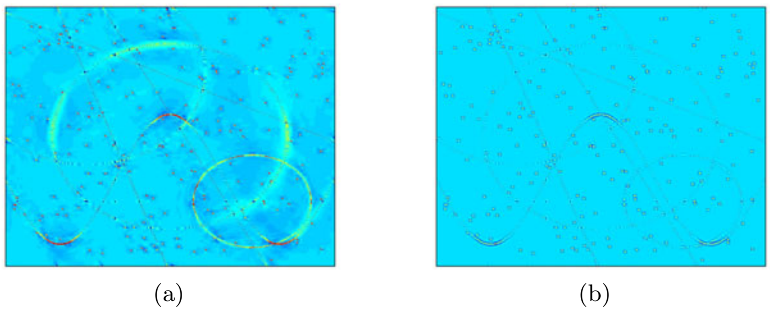


Fig. 2. (a) Pointlike image component extracted by MCALab without preprocessing. (b) Pointlike image component extracted by MCALab with preprocessing.

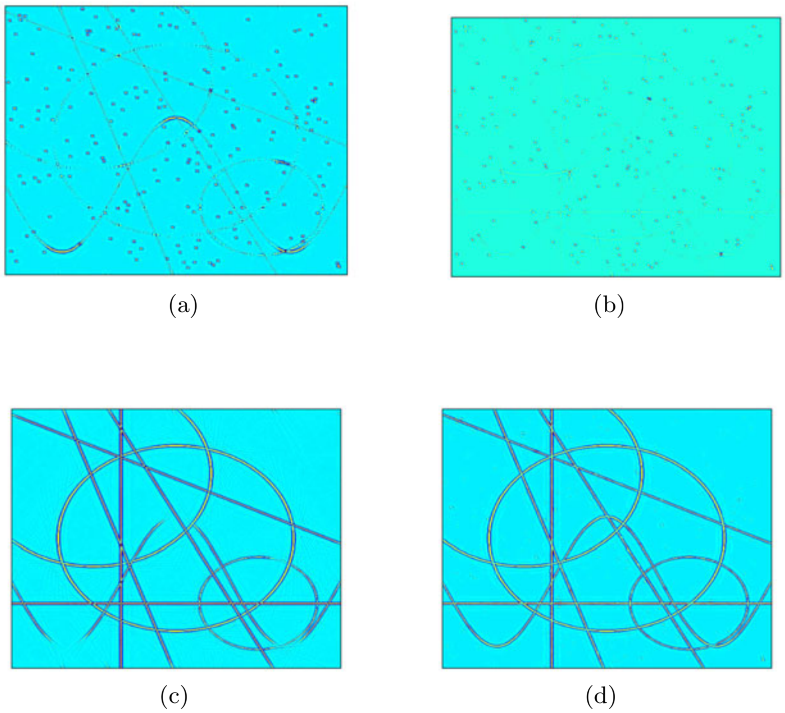
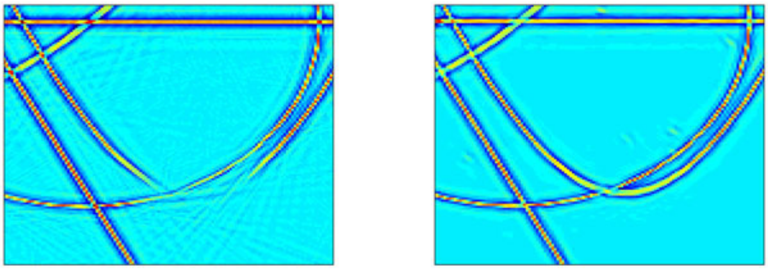


Fig. 3. (a) MCALab: Pointlike component. (b) Our scheme: Pointlike component. (c) MCALab: Curvelike component. (d) Our scheme: Curvelike component.

than (band-limited) curvelets, which positively affects the capturing of localized features of the curve as illustrated in Figure 3(d) and Figure 4(b). Hence, with respect to this visual comparison, our scheme outperforms MCALab, in particular, when curves with large curvature are present.



(a) (b)

Fig. 4. Zoomed images: (a) MCALab. (b) Our scheme

5.2 Comparison by Quantitative Measures

We now put the comparison on more solid ground by introducing two quantitative measures for analyzing how accurate our scheme as compared to MCALab extracts points and curves.

To define our first quantitative measure, let P and C be (binary) images containing points and curves, respectively, with image domain $\Omega \subset \mathbb{Z}^2$, and let $\hat{P}$ and $\hat{C}$ be the separated images from $I = P + C +$ noise by the separation scheme to be analyzed. Letting $T \geq 0$, B_T be defined by

$$\mathrm{B}_T(I) = \chi_{\{n \in \Omega : I(n) \geq T\}}, \quad \text{for a 2D image } I,$$

and g be a 2D discrete Gaussian filter, we introduce the test measures

$$M_p(\hat{P})(T) = \frac{\|g * P - g * (\mathrm{B}_{T \cdot \max(\hat{P})}(\hat{P}))\|_2}{\|g * P\|_2}$$

and

$$M_c(\hat{C})(T) = \frac{\|g * C - g * (\mathrm{B}_{T \cdot \max(\hat{C})}(\hat{C}))\|_2}{\|g * C\|_2}.$$

Using P and C as given by Figure 1, the graphs of the error functions $M_p(\hat{P})$ and $M_c(\hat{C})$ for our scheme and MCALab are plotted in Figure 5 depending on the threshold parameter $0 < T < 1$.

These figures imply that our scheme outperforms MCALab with respect to this quantitative measure.

As our second quantitative measure, we will use the running time of each scheme. With respect to this comparison measure, MCALab runs 182.19 sec (30 iterations) to produce the test results while our scheme takes 135.37 sec (15 iterations). The running time was computed by taking the average over 10 runs. Again, with respect to this measure our scheme performs superior.

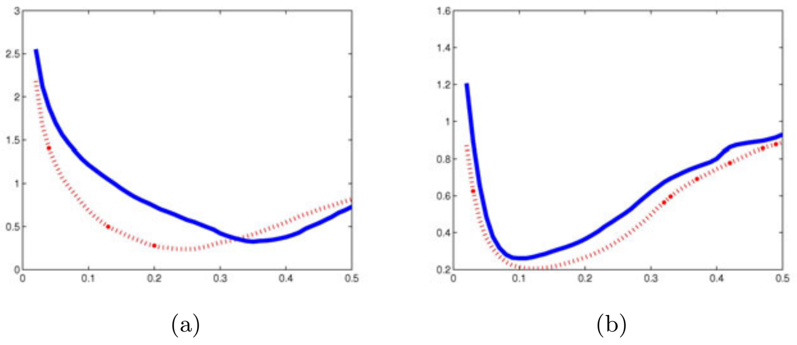


Fig. 5. (a) Graph of quantitative measure $T \mapsto M_p(\hat{P})(T)$: Our scheme (dashed curve) and MCALab (solid curve). (b) Graph of quantitative measure $T \mapsto M_p(\hat{C})(T)$: Our scheme (dashed curve) and MCALab (solid curve).

6 Application in Neurobiology

To test the performance of our scheme on real-world images, we apply it to an image of a neuron generated by fluorescence microscopy (Figure 6(a)), which is composed of 'spines' (pointlike features) and 'dendrites' (curvelike features). Figures 6(b) and (c) show the extracted images containing spines and dendrites, respectively.

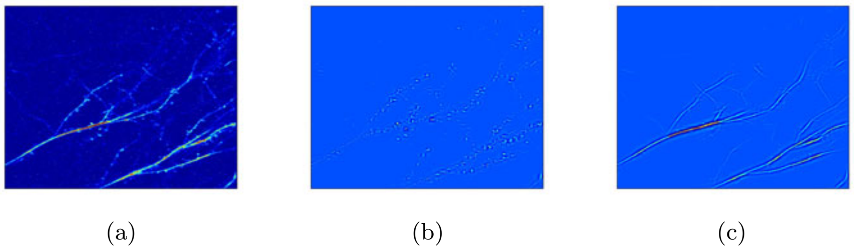


Fig. 6. (a) Image of neuron. (b) Extracted spines. (c) Extracted dendrites.

7 Conclusion

In this paper, we introduced a novel methodology for separating images into point- and curvelike features based on the new paradigm of sparse approximation and using ℓ_1 minimization. In contrast to other approaches, our algorithm utilizes a combined dictionary consisting of wavelets and shearlets, implemented as shift-invariant transforms, and is based on a mathematical theory. The excellent localization property of compactly supported shearlets allows shearlets to capture the curvelinear part very accurately and efficiently, even if the curvature is relatively large. Numerical results show that our scheme extracts point- and curvelike features more precise and uses less computing time than the state-of-the-art algorithm MCALab, which is based on wavelets and curvelets.

Acknowledgement. The first author would like to thank David Donoho and Michael Elad for inspiring discussions on related topics. We are also grateful to the research group by Roland Brandt for supplying the test image in Figure 6. The authors acknowledge support from DFG Grant SPP-1324, KU 1446/13. The first author also acknowledges partial support from DFG Grant KU 1446/14. We are particularly grateful to the two anonymous referees for their many valuable comments and suggestions which significantly improved this paper.

References

1. Candés, E.J., Donoho, D.L.: New tight frames of curvelets and optimal representations of objects with piecewise C^2 singularities. Comm. Pure and Appl. Math. 56, 216–266 (2004)
2. Donoho, D.L., Kutyniok, G.: Geometric Separation using a Wavelet-Shearlet Dictionary. In: Proc. of SampTA 2009, Marseille, France (2009)
3. Donoho, D.L., Kutyniok, G.: Microlocal analysis of the geometric separation problem (preprint)
4. Do, M.N., Vetterli, M.: The contourlet transform: An efficient directional multiresolution image representation. IEEE Trans. Image Process. 14, 2091–2106 (2005)
5. Donoho, D.L., Maleki, A., Shahram, M., Stodden, V., Ur-Rahman, I.: Fifteen years of reproducible research in computational harmonic analysis. Comput. Sci. Engrg. 11, 8–18 (2009)
6. Elad, M.: Sparse and redundant representations. Springer, New York (2010)
7. Fadilli, M.J., Starck, J.-L., Elad, M., Donoho, D.L.: MCALab: Reproducible research in signal and image decomposition and inpainting. IEEE Comput. Sci. Eng. Mag. 12, 44–63 (2010)
8. Guo, K., Kutyniok, G., Labate, D.: Sparse multidimensional representations using anisotropic dilation and shear operators. In: Wavelets and Splines, Athens, GA, pp. 189–201. Nashboro Press, Nashville (2006)
9. Kittipoom, P., Kutyniok, G., Lim, W.-Q: Construction of compactly supported shearlet frames. Constr. Approx. (to appear)
10. Kutyniok, G.: Sparsity equivalence of anisotropic decompositions (preprint)
11. Kutyniok, G., Lemvig, J., Lim, W.-Q: Compactly Supported Shearlets. In: Approximation Theory XIII. Springer, San Antonio (2010)
12. Kutyniok, G., Lim, W.-Q: Compactly supported shearlets are optimally sparse. J. Approx. Theory (to appear)
13. Kutyniok, G., Lim, W.-Q, Zhuang, X.: Digital Shearlet Transforms. In: Shearlets: Multiscale Analysis for Multivariate Data. Springer, New York (to appear)
14. Labate, D., Lim, W.-Q, Kutyniok, G., Weiss, G.: Sparse multidimensional representation using shearlets. In: Papadakis, M., Laine, A.F., Unser, M.A. (eds.) Proc. of SPIE Wavelets XI, vol. 5914, pp. 254–262. SPIE, Bellingham (2005)
15. Lim, W.-Q: The discrete shearlet transform: A new directional transform and compactly supported shearlet frames. IEEE Trans. Image Proc. 19, 1166–1180 (2010)

16. Lim, W.-Q: Nonseparable Shearlet Transforms (preprint)
17. Mallat, S.: A wavelet tour of signal processing. Academic Press, Inc., San Diego (1998)
18. Meyer, Y.: Oscillating Patterns in Image processing and nonlinear evolution equations. University Lecture Series, vol. 22. Amer. Math. Soc., Providence (2002)
19. Starck, J.-L., Elad, M., Donoho, D.: Image decomposition via the combination of sparse representation and a variational approach. IEEE Trans. Image Proc. 14, 1570–1582 (2005)

On a Special Class of Polynomial Surfaces with Pythagorean Normal Vector Fields

Miroslav Lávička and Jan Vršek

University of West Bohemia, Faculty of Applied Sciences, Department of Mathematics, Univerzitní 8, 301 00 Plzeň, Czech Republic
{lavicka,vrsekjan}@kma.zcu.cz

Abstract. Rational shapes with rational offsets, especially Pythagorean hodograph (PH) curves and Pythagorean normal vector (PN) surfaces, have been thoroughly studied for many years. However compared to PH curves, Pythagorean normal vector surfaces were introduced using dual approach only in their rational version and a complete characterization of polynomial surfaces with rational offsets, i.e., a polynomial solution of the well-known surface Pythagorean condition, still remains an open and challenging problem. In this contribution, we study a remarkable family of cubic polynomial PN surfaces with birational Gauss mapping, which represent a surface counterpart to the planar Tschirnhausen cubic. A full description of these surfaces is presented and their properties are discussed.

1 Introduction

Surfaces with Pythagorean normal vector fields (PN surfaces) were defined in [1] as a surface analogy to Pythagorean hodograph (PH) curves introduced in [2]. These surfaces provide an elegant and practical solution of various difficult problems occurring in technical applications, in particular in the context of CNC machining. They are distinguished by having rational offsets, i.e., tool paths do not have to be approximated and they are described exactly in NURBS form, which represents currently a universal standard in CAD. For a survey of shapes with Pythagorean property (i.e., possessing rational offsets) see [3] and references therein. Many interesting questions related to this subject have arisen, including analysis of geometric and algebraic properties of offsets, determining the number and type of their components and constructing suitable rational parameterizations, cf. [4,5,6,7,8,9].

Although PH curves in the plane and PN surfaces in the space share common goals, the main one being rationality of their offsets, one can find a significant difference between them. The curves with Pythagorean hodographs were introduced as planar polynomial shapes and a compact formula for their description based on Pythagorean triples of polynomials is available. Later, the concept was generalized also to rational PH curves, cf. [1], using a dual representation. The interplay between the different approaches to polynomial and rational PH curves was thoroughly studied in [10] and the former were established as a proper subset of the latter by presenting simple algebraic constraints. In addition, we would like to emphasize that a main advantage of polynomial PH curves is that their arc length is expressible as a polynomial function of the parameter (for rational PH curves the situation is more complicated as the integral of a rational function is not necessarily rational).

On the other hand, the situation in the surface case is absolutely different. We have only a description of rational Pythagorean normal vector surfaces reflecting again the dual approach and polynomial formulae (including an algebraic condition for the specialization of rational PN surfaces to polynomial ones) have not been known yet. Hence, finding a polynomial solution of the Pythagorean condition in the surface case still remains an open and challenging problem.

In this paper we will study an analogy between a polynomial PH curves and PN surfaces of low parametric degrees. As known, the only curve with a non-trivial quadratic polynomial parameterization is a parabola. This curve belongs to the so called LN curves (i.e., curves with Linear normals, cf. [11]) and thus it can be reparameterized coherrently with respect to the unit circle (yielding a rational PH parameterization) – a proper rational parameterization for the two-sided offset to a parabola was firstly constructed in [12]. The analogy in surface case was revealed quite recently in [13] and thoroughly studied in [14]. It was proved that all polynomial non-developable quadratic surfaces are again LN surfaces, i.e., as in the plane case they admit (after rational reparametrization) rational offsets. This property was used later in [15].

It is well-known fact that the unique polynomial PH cubic is the so-called Tschirnhausen cubic, cf. [2]. There also exists a whole affine family of LN cubics but they provide only rational PH parameterizations. On the other hand, the situation for cubic PN surfaces has not been answered yet. The only thing, one can find in literature, is a remark on a PN parameterization of the so called Enneper surface in [16,17]. This famous minimal surface is studied in [16] from the point of view isothermal (conformal) parameterizations which are closely related to Pythagorean-hodograph preserving mappings (cf. [18]). In this paper, we will follow this approach and show that there exists a whole class of cubic polynomial PN surfaces containing among others the mentioned Enneper surface. Moreover, we identify in this family a direct surface analogy to the Tschirnhausen cubic, i.e., a cubic PN surface of algebraic degree three.

The remainder of this paper is organized as follows. Section 2 recalls some basic facts concerning surfaces with Pythagorean normal vector fields, isothermal parameterizations and PH preserving mappings. Section 3 is devoted to a thorough study of cubic polynomial PN parameterizations. We present generators of these shapes and study them in connection to Enneper-Weierstrass formula for minimal surfaces. The geometric properties of cubic PN surfaces and curves on these surfaces are investigated in Section 4. Finally, we conclude the paper in Section 5.

2 Preliminaries

We briefly review fundamentals of rational surfaces with rational offsets and some related topics. The reader is referred to [3] and [1] for more details.

2.1 Rational Surfaces with Rational Offsets

Consider a parametric surface $\mathbf{x} : \mathbb{R}^2 \to \mathbb{R}^3$ given by a parameterization $\mathbf{x}(u,v)$. The *δ-offset* of $\mathbf{x}(u,v)$ is the set of all points in $\mathbb{R}^3$ that lie at a distance δ from $\mathbf{x}$. The two branches of the offset are given by

$$\mathbf{x}_\delta(u,v) = \mathbf{x}(u,v) \pm \delta\,\mathbf{N}(u,v), \qquad \mathbf{N}(u,v) = \frac{\mathbf{x}_1 \times \mathbf{x}_2}{\|\mathbf{x}_1 \times \mathbf{x}_2\|}, \tag{1}$$

where $\|\cdot\|$ denotes the usual Euclidean norm and $\mathbf{x}_1$ and $\mathbf{x}_2$ are partial derivatives with respect to u and v, respectively.

A study of offset rationality led to the class of *surfaces with Pythagorean normal vector fields* introduced in [1]. In this paper, we use a modified description (of course, equivalent with the original one). Let $\mathbf{x}(u,v)$ be a rational surface in $\mathbb{R}^n$, $n = 3,\dots$ for which there exists a rational function $\sigma(u,v) \in \mathbb{R}(u,v)$ such that

$$G(u,v) = \det(g_{ij}) = \sigma(u,v)^2, \quad i,j = 1,2, \tag{2}$$

where $G(u,v)$ is the associated Gram determinant with the entries $g_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j$. As the Gramian has for surfaces in $\mathbb{R}^3$ the following form

$$G(u,v) = g_{11}g_{22} - g_{12}^2 = \|\mathbf{x}_1 \times \mathbf{x}_2\|^2, \tag{3}$$

where g_{11}, g_{12}, g_{22} are the coefficients of the first fundamental form, then condition (2) shows that surfaces with *Pythagorean area elements* are in $\mathbb{R}^3$ equivalent to PN surfaces. Hence, (2) guarantees rational offsets of $\mathbf{x}$, cf. (1).

Let us recall that the approach based on the associated Gramian can be applied for all Pythagorean k-varieties, $1 \le k \le n-1$, in an arbitrary dimension n, specially for PH curves parameterized by $\mathbf{x}(t)$. These are distinguished by the property

$$G(t) = \det(g_{11}) = g_{11} = \|\mathbf{x}'(t)\|^2 = \sigma(t)^2. \tag{4}$$

For the sake of brevity, we will deal only with non-developable PN surfaces. Such surfaces $\mathbf{x}(u,v)$ can be obtained as the envelope of a two-parametric set of the associated tangent planes

$$T(u,v): \quad \mathbf{N}(u,v)\cdot\mathbf{x}(u,v) - h(u,v) = 0, \tag{5}$$

where $h(u,v) = e(u,v)/f(u,v)$ is a rational function (the so called *support function*) representing the oriented distance from the origin and $\mathbf{N}(u,v)$ is a rational parameterization of the unit sphere $\mathbb{S}^2$, cf. [1],

$$\mathbf{N}(u,v) = \left(\frac{2ac}{a^2+b^2+c^2}, \frac{2bc}{a^2+b^2+c^2}, \frac{a^2+b^2-c^2}{a^2+b^2+c^2}\right)^\top, \tag{6}$$

with $a = a(u,v)$, $b = b(u,v)$, $c = c(u,v)$ fulfilling the condition $\gcd(a,b,c) = 1$. A parametric representation of an arbitrary non-developable PN surface can be then obtained from the system of equations $T = 0$, $\partial T/\partial u = 0$, $\partial T/\partial v = 0$ using Cramer's rule – see formula (3.3) in [1].

We will consider only such $a(u,v)$, $b(u,v)$, $c(u,v)$ in what follows that (6) is a proper parameterization of $\mathbb{S}^2$. The corresponding PN surfaces with birational Gauss map play the most prominent role among all PN parameterizations as they are closed under the operation of convolution and moreover, they preserve the PN property when the convolution with any arbitrary PN surface is constructed, cf. [17] for more details.

2.2 Isothermal Surfaces and Pythagorean-Hodograph Preserving Mappings

A parametric surface $\mathbf{x}(u,v)$ is said to be *isothermal* if the coefficients of the first fundamental form satisfy the condition

$$g_{11} = g_{22}, \quad g_{12} = 0. \tag{7}$$

The parameter lines on a isothermal surface are orthogonal and their speeds are equal. Clearly, all polynomial or rational isothermal surfaces are PN surfaces, cf. (3).

Moreover, isothermal surfaces considered as mappings $\mathbf{x} : \mathbb{R}^2 \to \mathbb{R}^3$ are closely related to the class of the so called *Pythagorean-hodograph preserving mappings* distinguished by the property that for every rational PH curve $\mathbf{r}(t) = (u(t), v(t))^\top$, the image $\mathbf{x}(\mathbf{r}(t))$ is also PH. More precisely if $g_{11} = g_{22} = \varrho(u,v)^2$, $g_{12} = 0$ then the surface $\mathbf{x}(u,v)$ is called a *scaled* Pythagorean-hodograph preserving mapping, cf. [18] for more details. Obviously, all parameter lines on such PN surfaces are rational (not necessarily planar) PH curves. In [16], the Enneper surface is studied as a particular example of PH preserving isothermal surfaces.

2.3 A Note on Polynomial Minimal Surfaces

Any simply connected *minimal surface* in $\mathbb{R}^3$ (i.e., a surface with zero mean curvature) can be represented by the Enneper-Weierstrass parameterization

$$\mathbf{x}(u,v) = \begin{pmatrix} \mathfrak{Re}\left(\int_0^\omega f(z)(1-g(z)^2)dz\right) \\ \mathfrak{Re}\left(\int_0^\omega if(z)(1+g(z)^2)dz\right) \\ \mathfrak{Re}\left(\int_0^\omega 2f(z)g(z)dz\right) \end{pmatrix}, \quad \omega = u + iv, \tag{8}$$

where f is a holomorphic function on a domain D, g is meromorphic on D and the product fg^2 is holomorphic on D. Then, the coefficients of the first fundamental form are given by

$$g_{11} = g_{22} = \left(|f|(|g|^2+1)\right)^2 = \varrho(u,v)^2, \quad g_{12} = 0. \tag{9}$$

Thus, all polynomial minimal surfaces are isothermal (i.e., PN) with the PH preserving property.

Consequently, (8) can be easily used for generating polynomial PN surfaces. As a particular example, choosing $f(z) = 1$, $g(z) = z$ we obtain the well-known parameterization of the Enneper surface

$$\mathbf{x}(u,v) = \left(u - \frac{u^3}{3} + uv^2, -v - u^2v + \frac{v^3}{3}, u^2 - v^2\right)^\top, \tag{10}$$

possessing Tschirnhausen cubics as parameter lines, see Fig 1.

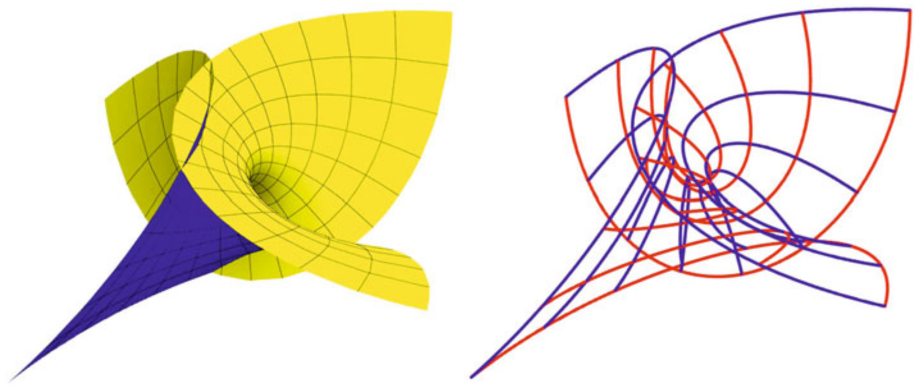


Fig. 1. The Enneper surface $\mathbf{P}_1$ (left) and its parameter lines, i.e., the Tschirnhausen cubics (right)

3 Polynomial Cubic PN surfaces

In this section, we will focus on cubic polynomial PN surfaces with birational Gauss mapping, which represent a surface counterpart to the Tschirnhausen cubic. A full description of these surfaces is given and their properties are studied.

3.1 Cubic PN Parameterizations

Similarly to the curve case, the most simple non-trivial polynomial PN parameterizations are of the parametric degree 3. Let $\mathbf{x}(u,v)$ be a parameterization of this type. The degree of the associated normal vector field (6) is even and after computing the dual representation (and comparing the degrees of polynomials) we can conclude that this degree is for cubic PN surfaces at most 4.

Furthermore, it can be shown that if $\mathbf{x}(u,v)$ is a cubic polynomial PN parameterization then the associated unit normal vector field (6) forms a birational parameterization of the unit sphere. Thus any cubic polynomial PN parameterizations may be obtained by choosing some linear polynomials $a(u,v)$, $b(u,v)$, $c(u,v)$ and a suitable support function $h(u,v) = e/(a^2+b^2+c^2)$, where $e(u,v)$ is a polynomial. By a simple reparameterization we arrive at

$$a(u,v) = u, \quad b(u,v) = v, \quad c(u,v) = pu + qv + 1, \qquad p,q \in \mathbb{R}, \tag{11}$$

and thus w.l.o.g. we consider (11) in what follows.

First, we focus on the special but the most interesting case $p = q = 0$. After multiplying (5) by $a^2+b^2+c^2 = u^2+v^2+1$ we arrive at

$$2ux(u,v) + 2vy(u,v) + (u^2+v^2-1)z(u,v) = e(u,v). \tag{12}$$

The cubic parameterization $\mathbf{x}(u,v) = \big(x(u,v), y(u,v), z(u,v)\big)^\top$ may be then obtained by solving the following system of equations

$$\mathbf{M}\cdot\mathbf{x} = \begin{pmatrix} 2u & 2v & u^2+v^2-1 \\ 2 & 0 & 2u \\ 0 & 2 & 2v \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} e(u,v) \\ \dfrac{\partial\, e(u,v)}{\partial\, u} \\ \dfrac{\partial\, e(u,v)}{\partial\, v} \end{pmatrix} = \mathbf{E}(u,v). \tag{13}$$

Then,

$$\mathbf{x}(u,v) = \mathbf{M}^{-1}\cdot\mathbf{E}, \quad \text{or} \quad \mathbf{x}(u,v) = \frac{1}{\det\mathbf{M}}\mathbf{M}^{A}\cdot\mathbf{E}, \tag{14}$$

where

$$\det\mathbf{M} = -4(u^2+v^2+1) \tag{15}$$

and the adjoint matrix to $\mathbf{M}$ has the form

$$\mathbf{M}^{A} = \begin{pmatrix} -4u & 2(u^2-v^2-1) & 4uv \\ -4v & 4uv & 2(-u^2+v^2-1) \\ 4 & -4u & -4v \end{pmatrix}. \tag{16}$$

Hence, we are looking for all polynomials $e(u,v)$ such that

$$\det(\mathbf{M})x_i = \mathbf{M}^{A}_{i1}e(u,v) + \mathbf{M}^{A}_{i2}\frac{\partial\, e(u,v)}{\partial\, u} + \mathbf{M}^{A}_{i3}\frac{\partial\, e(u,v)}{\partial\, v}, \quad i = 1,2,3, \tag{17}$$

simultaneously assuming

$$\max\big\{\deg(x(u,v)), \deg(y(u,v)), \deg(z(u,v))\big\} = 3. \tag{18}$$

This leads to a system of linear equations with unknowns being coefficients of the polynomial $e(u,v)$. It can be shown (for the sake of brevity we omit this purely technical step) that all the cubic PN parameterizations gained by this computation have the following form

$$\mathbf{x}(u,v) = \alpha_1\mathbf{P}_1 + \alpha_2\mathbf{P}_2 + \alpha_3\mathbf{P}_3 + (\tau_1,\tau_2,\tau_3)^\top, \tag{19}$$

where

$$\begin{aligned} \mathbf{P}_1 &= \left(u - \frac{u^3}{3} + uv^2, -v - u^2v + \frac{v^3}{3}, u^2 - v^2\right)^\top, \\ \mathbf{P}_2 &= \left(-v + u^2v - \frac{v^3}{3}, -u - \frac{u^3}{3} + uv^2, -2uv\right)^\top, \\ \mathbf{P}_3 &= \left(u - \frac{u(u^2+v^2)}{3}, v - \frac{v(u^2+v^2)}{3}, u^2+v^2\right)^\top, \end{aligned} \tag{20}$$

and $(\alpha_1,\alpha_2,\alpha_3) \in \mathbb{R}^3 \setminus (0,0,0)$, $\tau_1,\tau_2,\tau_3 \in \mathbb{R}$. The above introduced surfaces $\mathbf{P}_1$, $\mathbf{P}_2$, $\mathbf{P}_3$ (generators of an arbitrary cubic PN parameterization of the first type) will be studied in more detail in the following subsections.

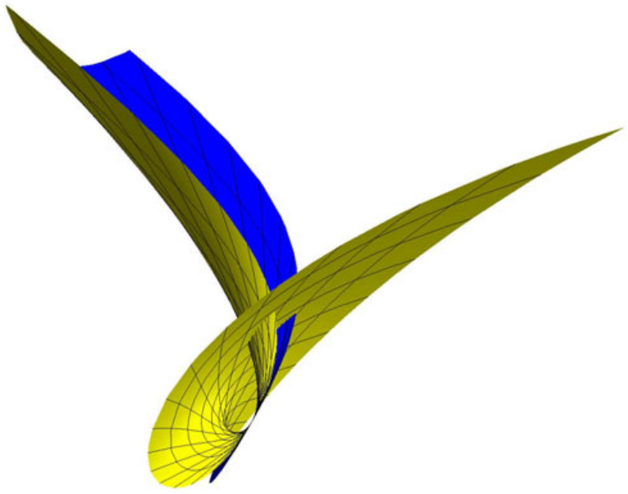


Fig. 2. A cubic PN surface of the second type $\left(p = q = \frac{1}{2}\right)$

An analogous approach can also be applied when the polynomial $c(u, v)$ has the form $pu + qv + 1$, cf. (11). If both coefficients p, q do not vanish simultaneously we arrive at the unique solution (up to a translation and scaling) in the form

$$\begin{pmatrix} (pu+qv)(u^2p^3+2vup^2q+3p^2u-pu^2+v^2q^2p+v^2p+3pqv+3p-2uqv) \\ (pu+qv)(u^2p^2q+2vuq^2p+3puq-2puv+qu^2+q^3v^2+3q^2v-qv^2+3q) \\ (pu+qv)^2(2pu+2qv+3) \end{pmatrix}. \quad (21)$$

One example of this type of PN surfaces is given in Fig. 2. However, as for any fixed choice of p, q (i.e., for a fixed normal vector field) one can construct only one surface (up to a translation and scaling) we will not deal with these surfaces in what follows and focus mainly on the first family described by (19).

3.2 Enneper Minimal Surface and Its PN Parameterizations

To get a better insight into the properties of cubic PN surfaces from the family (19), we will thoroughly study the generating surfaces $\mathbf{P}_i$, $i = 1, 2, 3$. Obviously, the surface $\mathbf{P}_1(u, v)$ is nothing else than the Enneper surface (10) – some of its properties were recalled in the previous subsections.

Now, we will study the parameterization $\mathbf{P}_2(u, v)$. Computing the coefficients of the first fundamental form

$$g_{11} = g_{22} = (u^2 + v^2 + 1)^2, \quad g_{12} = 0, \tag{22}$$

we conclude that this surface is again isothermal with PH preserving property (its parameter lines are spatial PH cubics), see Fig 3. Furthermore, $\mathbf{P}_2(u, v)$ is a polynomial

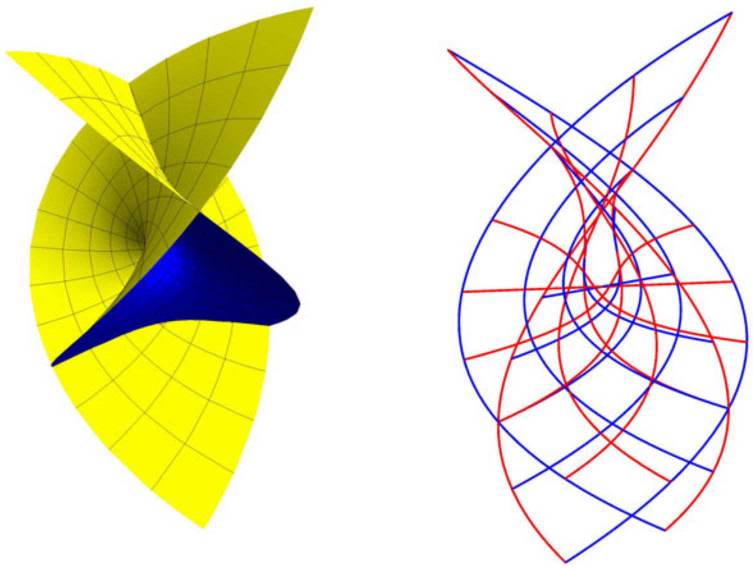


Fig. 3. Enneper surface $\mathbf{P}_2$ (left) and its parameter lines (right)

minimal surface and thus it may be given by the Enneper-Weierstrass formula ([8]) for

$$f(z) = i, \quad g(z) = z. \tag{23}$$

In addition, $\mathbf{P}_2(u, v)$ is a surface of the algebraic degree 9 and if $F(x, y, z) = 0$ is its implicit equation then $F(x', y', z') = 0$, where

$$\begin{aligned} x &= \frac{\sqrt{2}}{2}x' - \frac{\sqrt{2}}{2}y', \\ y &= \frac{\sqrt{2}}{2}x' + \frac{\sqrt{2}}{2}y', \\ z &= z', \end{aligned} \tag{24}$$

is the implicit equation of $\mathbf{P}_1(u, v)$. Hence, $\mathbf{P}_2(u, v)$ is again a parameterization of the Enneper surface (rotated by $\frac{\pi}{4}$ along the z-axis).

Finally, if we restrict our considerations only to the class $\alpha_1\mathbf{P}_1(u, v) + \alpha_2\mathbf{P}_2(u, v)$ then any representative is a polynomial minimal surface described by the Enneper-Weierstrass formula ([8]) for

$$f(z) = \alpha_1 + \alpha_2 i, \quad g(z) = z. \tag{25}$$

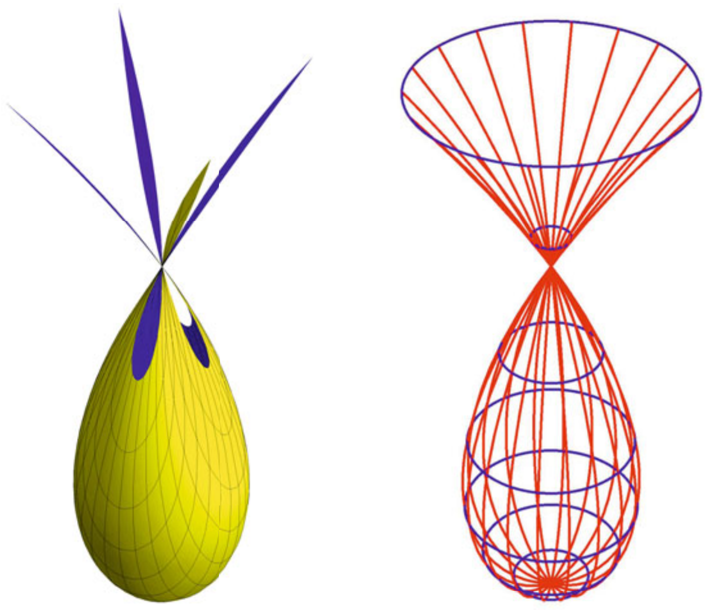


Fig. 4. Tschirnhausen cubic surface $\mathbf{P}_3(u,v)$ (left) and the parameter lines of the rational parameterization constructed by the rotation of the Tschirnhausen cubic curve (right)

All of these parameterized surfaces are Enneper surfaces with parameter lines being (generally) spatial PH cubics characterized by the associated Gramians, cf. (4),

$$\begin{aligned} G(u) &= (\alpha_1^2+\alpha_2^2)(1+u^2+v_0^2)^2, \\ G(v) &= (\alpha_1^2+\alpha_2^2)(1+u_0^2+v^2)^2. \end{aligned} \tag{26}$$

In particular, planar parameter lines (i.e., Tschirnhausen cubics) are obtained only when $\alpha_2 = 0$.

3.3 Tschirnhausen Cubic Surface

In this subsection, we will study the third generating surface $\mathbf{P}_3$ which is not (compared to the surfaces $\mathbf{P}_1$ and $\mathbf{P}_2$) minimal and its parameter lines are not PH curves. This surface possesses the algebraic degree 3 and is described by the implicit equation

$$9x^2 + 9y^2 + z(z+3)^2 = 0. \tag{27}$$

It is an affine image of the surface known as *Ding-dong surface* mentioned among interesting cubic surfaces of revolution, see Fig. 4.

Despite the fact that its parameter lines are not PH curves, this surface is a direct surface analogy to the Tschirnhausen cubic as it is not only a polynomial cubic

Pythagorean variety but it also has the algebraic degree three. However, it can be obtained as a rotational surface given by the meridian (for $y = 0$, i.e., setting $v = 0$ into the parameterization $\mathbf{P}_3(u, v)$)

$$\left(u - \frac{u^3}{3}, u^2\right)^\top, \tag{28}$$

which is the Tschirnhausen cubic, cf. Fig.4 (right). Nevertheless, the parameterization reflecting the rotational nature of this surface is not polynomial but rational.

Furthermore, if we compute the 2-valued *support functions* of both Tschirnhausen cubic varieties (see e.g. [19,20,21] for more details about the support function representation theory) then we find another analogy. For the Tschirnhausen cubic curve we obtain

$$h(n_1, n_2) = \frac{3n_1^2 n_2 + 2n_2^3 \pm 2\sqrt{(n_1^2 + n_2^2)^3}}{3n_1^2} = \frac{3n_2 - n_2^3 \pm 2}{3(1 - n_2^2)}, \tag{29}$$

where $n_1^2 + n_2^2 = 1$, and for the Tschirnhausen cubic surface we arrive at

$$h(n_1, n_2, n_3) = \frac{3(n_1^2 + n_2^2)n_3 + 2n_3^3 \pm 2\sqrt{(n_1^2 + n_2^2 + n_3^2)^3}}{3(n_1^2 + n_2^2)} = \frac{3n_3 - n_3^3 \pm 2}{3(1 - n_3^2)}, \tag{30}$$

where $n_1^2 + n_2^2 + n_3^2 = 1$.

Obviously, all cubically parameterized PN surfaces represent a subfamily of Bézier cubic surfaces, cf. [22]. A Bézier triangular cubic patch on the Tschirnhausen surface is shown in Fig. 5 – its boundary lines are planar (but generally not PH) cubics.

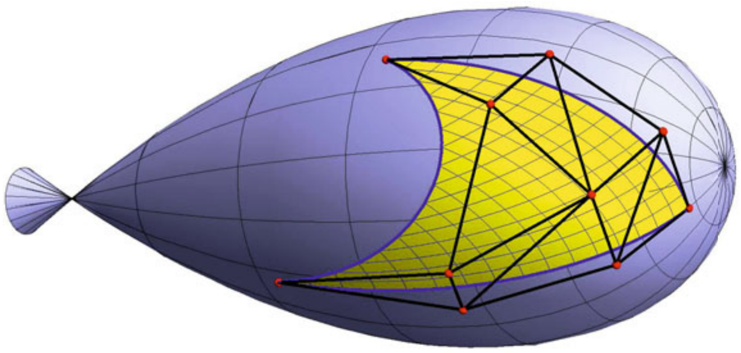


Fig. 5. A triangular Bézier patch on the Tschirnhausen cubic surface

4 Dual Kinematic Description

Consider the unit normal vector field (6) for $a = u$, $b = v$, $c = 1$. Next, let be given two focal parabolas

$$\mathbf{p}_1(u) = \left(2u, 0, u^2 - \frac{1}{2}\right)^\top \quad \text{and} \quad \mathbf{p}_2(v) = \left(0, -2v, -v^2 + \frac{1}{2}\right)^\top. \tag{31}$$

As it holds

$$\mathbf{N}(u,v) = \frac{\mathbf{p}_1(u) - \mathbf{p}_2(v)}{\|\mathbf{p}_1(u) - \mathbf{p}_2(v)\|} = \frac{\left(2u, 2v, u^2 + v^2 - 1\right)^\top}{(u^2 + v^2 + 1)}, \tag{32}$$

this gives us a geometric description of the normal vector field of all polynomial cubic PN surfaces (19).

In particular, it is well known fact that the envelope of two-parameter family of the planes of symmetry of two points $\mathbf{p}_1(u)$ and $\mathbf{p}_2(v)$ is the Enneper surface, see [16]. To generalize this construction we consider translated copies of these planes in the direction of their normal vectors, i.e., we take the family of tangent planes

$$\left(\mathbf{p}_1(u) - \mathbf{p}_2(v)\right) \cdot \left(\mathbf{x} + \frac{\lambda}{1-\lambda}\mathbf{p}_2 - \frac{1}{1-\lambda}\mathbf{p}_1(u)\right) = 0, \tag{33}$$

where $\lambda \in \mathbb{R}$ is the division ratio describing the position of a chosen point on the line $\mathbf{p}_1\mathbf{p}_2$.

All envelope surfaces given by (33) are polynomial cubic PN surfaces given by the parameterization

$$\mathbf{P}(u,v) = \frac{1}{1-\lambda}\begin{pmatrix} u(-u^2 + v^2 - 2\lambda v^2 + 3) \\ v(\lambda u^2 - 2u^2 - \lambda v^2 + 3\lambda) \\ (6u^2 + 6\lambda v^2 - \lambda - 1)/2 \end{pmatrix}. \tag{34}$$

As the factor $1/(1-\lambda)$ brings only scaling, we can omit it in what follows.

Then after a short computation one can express the envelope surfaces (34) in the form of the linear combination (19) for the values

$$\alpha_1 = \frac{3}{2}(1-\lambda),\ \alpha_2 = 0,\ \alpha_3 = \frac{3}{2}(1+\lambda), \quad \tau_1 = \tau_2 = 0,\ \tau_3 = -\frac{1}{2}(1+\lambda). \tag{35}$$

Hence, especially for $\lambda = -1$ we obtain an envelope of planes of symmetry and thus (up to scaling) the Enneper surface $\mathbf{P}_1(u,v)$ (see Fig. 6), the case $\lambda = 1$ is the limit case and the resulting surface is (up to scaling and translation) the Tschirnhausen cubic surface $\mathbf{P}_3(u,v)$. Finally, all the surfaces (34) are in a direct correspondence with surfaces from the family $\alpha_1\mathbf{P}_1 + \alpha_3\mathbf{P}_3$, see Fig. 7 for one particular example different from the canonical surfaces $\mathbf{P}_1$, $\mathbf{P}_3$.

It is also seen that the remaining PN surfaces from the class (19) (i.e., surfaces with $\alpha_2 \neq 0$) cannot be obtained via (34). To gain all surfaces of this type one would have to allow also non-constant $\lambda(u,v)$.

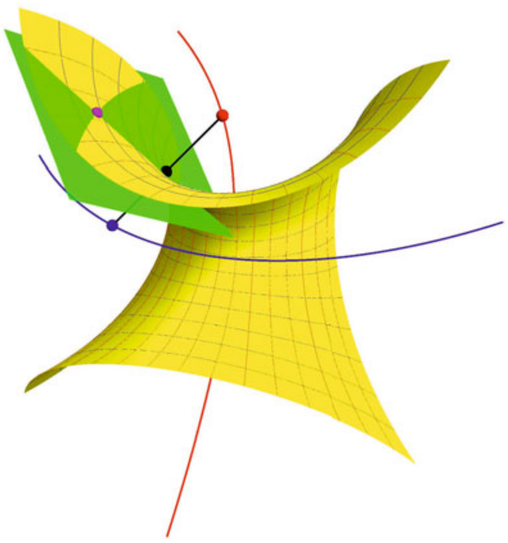


Fig. 6. A kinematic description of the Enneper surface $\mathbf{P}_1(u,v)$. The focal parabolas (red and blue) with the points $\mathbf{p}_1, \mathbf{p}_2$ and the associated tangent plane (green) touching the Enneper surface (at the magenta point).

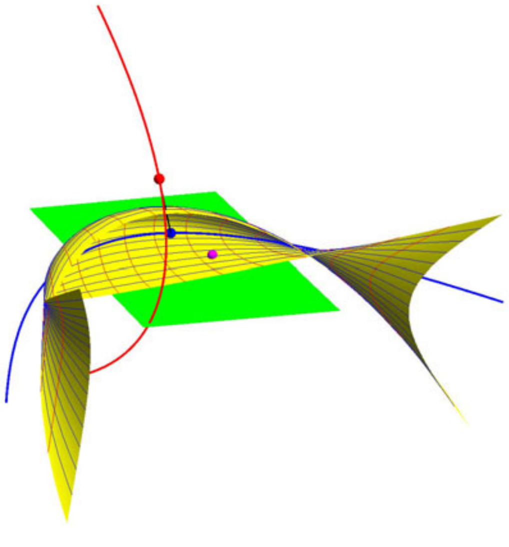


Fig. 7. Kinematic description of a general PN surface given dually by (33). The focal parabolas (red and blue) with the points $\mathbf{p}_1$, $\mathbf{p}_2$ and the associated tangent plane (green) touching the PN surface (at the magenta point) .

5 Conclusion

In the present paper we continued the discussion from [16] and extended the ideas concerning the Enneper surface to further cubic polynomial PN parameterizations. This study was motivated by the fact that a polynomial solution of the Pythagorean condition in the surface case remains an open problem and a general formula is still unknown.

Especially, we studied a remarkable class of cubic parametric surfaces with Pythagorean normals. Full description of these surfaces was given and some of their properties, e.g. their dual kinematic construction, were presented. In addition, the close relation to isothermal surfaces and PH preserving mappings was also studied.

Acknowledgement. The work on this paper was supported by the Research Plan MSM 4977751301. We thank all referees for their comments, which helped us to improve the paper.

References

1. Pottmann, H.: Rational curves and surfaces with rational offsets. Computer Aided Geometric Design 12(2), 175–192 (1995)
2. Farouki, R., Sakkalis, T.: Pythagorean hodographs. IBM Journal of Research and Development 34(5), 736–752 (1990)
3. Farouki, R.: Pythagorean-Hodograph Curves: Algebra and Geometry Inseparable. Springer, Heidelberg (2008)
4. Lü, W., Pottmann, H.: Rational parameterization of quadrics and their offsets. Computing 57, 135–147 (1996)
5. Arrondo, E., Sendra, J., Sendra, J.R.: Parametric generalized offsets to hypersurfaces. Journal of Symbolic Computation 23, 267–285 (1997)
6. Arrondo, E., Sendra, J., Sendra, J.R.: Genus formula for generalized offset curves. Journal of Pure and Applied Algebra 136, 199–209 (1999)
7. Maekawa, T.: An overview of offset curves and surfaces. Computer-Aided Design 31, 165–173 (1999)
8. Sendra, J.R., Sendra, J.: Algebraic analysis of offsets to hypersurfaces. Mathematische Zeitschrift 237, 697–719 (2000)
9. Vršek, J., Lávička, M.: On convolution of algebraic curves. Journal of symbolic computation 45(6), 657–676 (2010)
10. Farouki, R.T., Pottmann, H.: Polynomial and rational Pythagorean-hodograph curves reconciled. In: Proceedings of the 6th IMA Conference on the Mathematics of Surfaces, pp. 355–378. Clarendon Press, New York (1996)
11. Jüttler, B.: Triangular Bézier surface patches with linear normal vector field. In: Cripps, R. (ed.) The Mathematics of Surfaces VIII. Information Geometers, pp. 431–446 (1998)
12. Lü, W.: Offset-rational parametric plane curves. Comput. Aided Geom. Des. 12(6), 601–616 (1995)
13. Lávička, M., Bastl, B.: Rational hypersurfaces with rational convolutions. Computer Aided Geometric Design 24(7), 410–426 (2007)
14. Peternell, M., Odehnal, B.: Convolution surfaces of quadratic triangular Bézier surfaces. Computer Aided Geometric Design 25, 116–129 (2008)
15. Bastl, B., Jüttler, B., Kosinka, J., Lávička, M.: Computing exact rational offsets of quadratic triangular Bézier surface patches. Computer-Aided Design 40, 197–209 (2008)

16. Ueda, K.: Pythagorean-hodograph curves on isothermal surfaces. In: Cripps, R. (ed.) The Mathematics of Surfaces. VIII. Proceedings of the 8th IMA Conference Held in Birmingham, GB, pp. 339–353. Information Geometers, Winchester (1998)
17. Lávička, M., Bastl, B.: PN surfaces and their convolutions with rational surfaces. Computer Aided Geometric Design 25, 763–774 (2008)
18. Kim, G.I., Lee, S.: Pythagorean-hodograph preserving mappings. J. Comput. Appl. Math. 216(1), 217–226 (2008)
19. Gravesen, J., Jüttler, B., Šír, Z.: On rationally supported surfaces. Computer Aided Geometric Design 25, 320–331 (2008)
20. Šír, Z., Gravesen, J., Jüttler, B.: Curves and surfaces represented by polynomial support functions. Theoretical Computer Science 392(1-3), 141–157 (2008)
21. Lávička, M., Bastl, B., Šír, Z.: Reparameterization of curves and surfaces with respect to their convolution. In: Dæhlen, M., Floater, M., Lyche, T., Merrien, J.-L., Mørken, K., Schumaker, L.L. (eds.) MMCS 2008. LNCS, vol. 5862, pp. 285–298. Springer, Heidelberg (2010)
22. Farin, G.: Curves and surfaces for CAGD: a practical guide. Morgan Kaufmann Publishers Inc., San Francisco (2002)

Convergence Rate of the Causal Jacobi Derivative Estimator

Da-yan Liu[1,2], Olivier Gibaru[1,3], and Wilfrid Perruquetti[1,4]

[1] INRIA Lille-Nord Europe, Équipe Projet Non-A, Parc Scientifique de la Haute Borne 40, avenue Halley Bât. A, Park Plaza, 59650 Villeneuve d'Ascq, France

[2] Université de Lille 1, Laboratoire de Paul Painlevé, 59650, Villeneuve d'Ascq, France

dayan.liu@inria.fr

[3] Arts et Metiers ParisTech centre de Lille, Laboratory of Applied Mathematics and Metrology (L2MA), 8 Boulevard Louis XIV, 59046 Lille Cedex, France

olivier.gibaru@ensam.eu

[4] École Centrale de Lille, Laboratoire de LAGIS, BP 48, Cité Scientifique, 59650 Villeneuve d'Ascq, France

wilfrid.perruquetti@inria.fr

Abstract. Numerical causal derivative estimators from noisy data are essential for real time applications especially for control applications or fluid simulation so as to address the new paradigms in solid modeling and video compression. By using an analytical point of view due to Lanczos [9] to this causal case, we revisit n^{th} order derivative estimators originally introduced within an algebraic framework by Mboup, Fliess and Join in [14,15]. Thanks to a given noise level δ and a well-suitable integration length window, we show that the derivative estimator error can be $\mathcal{O}(\delta^{\frac{q+1}{n+1+q}})$ where q is the order of truncation of the Jacobi polynomial series expansion used. This so obtained bound helps us to choose the values of our parameter estimators. We show the efficiency of our method on some examples.

Keywords: Numerical differentiation, Ill-posed problems, Jacobi orthogonal series.

1 Introduction

There exists a large class of numerical derivative estimators which were introduced according to different scopes ([8,18,1,22,20,16,17]). When the initial discrete data are corrupted by a noise, numerical differentiation becomes an ill-posed problem. By using an algebraic method inspired by [6,13,10], Mboup, Fliess and Join introduced in [14,15] real-time numerical differentiation by integration estimators that provide an effective response to this problem. Concerning the robustness of this method, [4,5] give more theoretical foundations. These estimators extend those introduced by [9,19,23] in the sense that they use Jacobi polynomials. In [14], the authors show that the mismodelling due to the truncation of the Jacobi expansion can be improved by allowing a small time-delay in the derivative estimation. This time-delay is obtained as the product of the length of the integration window by the smallest root of the first Jacobi polynomial in the remainder of series expansion.

In [11], we extend to the real domain the parameter values of these Jacobi estimators. This allows us to decrease the value of this smallest root and consequently the time-delay estimation. In [12], we study for center derivative Jacobi estimators the convergence rate of these estimators.

Thanks to these results, we propose in this article to tackle the causal convergence rate case. We give an optimal convergence rate of these estimators depending on the derivative order, the noise level of the data and the truncation order. Moreover, we show that the estimators for the n^{th} order derivative of a smooth function can be obtained by taking n derivations to the zero-order estimator of the function. Hence, we can give a simple expression for these estimators, which is much easier to calculate than the one given in [12].

This paper is organized as follows: in Section 2 the causal estimators introduced in [14] are studied with extended parameters. The convergence rate of these estimators are then studied. Finally, numerical tests are given in Section 3. They help us to show the efficiency and the stability of this proposed estimators. We will see that the numerical integration error may also reduce this time-delay for a special class of functions.

2 Derivative Estimations by Using Jacobi Orthogonal Series

Let $f^{\delta} = f + \varpi$ be a noisy function defined on an open interval $I \subset \mathbb{R}$, where $f \in \mathcal{C}^n(I)$ with $n \in \mathbb{N}$ and ϖ be a noise[1] which is bounded and integrable with a noise level δ, *i.e.* $\delta = \sup\limits_{x \in I} |\varpi(x)|$. Contrary to [19] where Legendre polynomials were used, we propose to use, as in [14,15], truncated Jacobi orthogonal series so as to estimate the n^{th} order derivative of f. In this section, we are going to give a family of causal Jacobi estimators by using Jacobi polynomials defined on $[0,1]$. From now on, we assume that the parameter $h > 0$ and we denote $I_h := \{x \in I; [x-h, x] \subset I\}$.

The n^{th} order Jacobi polynomials (see [21]) defined on $[0,1]$ are defined as follows

$$P_n^{(\alpha,\beta)}(t) = \sum_{j=0}^{n} \binom{n+\alpha}{j} \binom{n+\beta}{n-j} (t-1)^{n-j} (t)^j \tag{1}$$

where $\alpha, \beta \in]-1, +\infty[$. Let g_1 and g_2 be two functions which belong to $\mathcal{C}([0,1])$, then we define the scalar product of these functions by

$$\langle g_1(\cdot), g_2(\cdot) \rangle_{\alpha,\beta} := \int_0^1 w_{\alpha,\beta}(t) g_1(t) g_2(t) dt,$$

[1] More generally, the noise is a stochastic process, which is bounded with certain probability and integrable in the sense of convergence in mean square (see [11]).

where $w_{\alpha,\beta}(t) = (1-t)^{\alpha} t^{\beta}$ is a weighted function. Hence, we can denote its associated norm by $\| \cdot \|_{\alpha,\beta}$. We then have

$$\|P_n^{(\alpha,\beta)}\|_{\alpha,\beta}^2 = \frac{1}{2n+\alpha+\beta+1} \frac{\Gamma(\alpha+n+1)\Gamma(\beta+n+1)}{\Gamma(\alpha+\beta+n+1)\Gamma(n+1)}. \tag{2}$$

Let us recall two useful formulae (see [21])

$$P_n^{(\alpha,\beta)}(t) w_{\alpha,\beta}(t) = \frac{(-1)^n}{n!} \frac{d^n}{dt^n}[w_{\alpha+n,\beta+n}(t)] \quad \text{(the Rodrigues formula)}, \tag{3}$$

$$\frac{d}{dt}[P_n^{(\alpha,\beta)}(t)] = (n+\alpha+\beta+1) P_{n-1}^{(\alpha+1,\beta+1)}(t). \tag{4}$$

Let us ignore the noise ϖ for a moment. Since f is assumed to belong to $\mathcal{C}^n(I)$, we define the q^{th} ($q \in \mathbb{N}$) order truncated Jacobi orthogonal series of $f^{(n)}(x-ht)$ ($t \in [0,1]$) by the following operator: $\forall x \in I_h$,

$$D_{h,\alpha,\beta,q}^{(n)} f(x-th) := \sum_{i=0}^{q} \frac{\left\langle P_i^{(\alpha+n,\beta+n)}(\cdot), f^{(n)}(x-h\cdot) \right\rangle_{\alpha+n,\beta+n}}{\|P_i^{(\alpha+n,\beta+n)}\|_{\alpha+n,\beta+n}^2} \, P_i^{(\alpha+n,\beta+n)}(t). \tag{5}$$

We also define the $(q+n)^{th}$ order truncated Jacobi orthogonal series of $f(x-ht)$ ($t \in [0,1]$) by the following operator

$$\forall x \in I_h,\ D_{h,\alpha,\beta,q}^{(0)} f(x-th) := \sum_{i=0}^{q+n} \frac{\left\langle P_i^{(\alpha,\beta)}(\cdot), f(x-h\cdot) \right\rangle_{\alpha,\beta}}{\|P_i^{(\alpha,\beta)}\|_{\alpha,\beta}^2} \, P_i^{(\alpha,\beta)}(t). \tag{6}$$

It is easy to show that for each fixed value x, $D_{h,\alpha,\beta,q}^{(0)} f(x-h\cdot)$ is a polynomial which approximates the function $f(x-h\cdot)$. We can see in the following lemma that $D_{h,\alpha,\beta,q}^{(n)} f(x-h\cdot)$ is in fact connected to the n^{th} order derivative of $D_{h,\alpha,\beta,q}^{(0)} f(x-h\cdot)$. It can be expressed as an integral of f.

Lemma 1. *Let $f \in \mathcal{C}^n(I)$, then we have*

$$\forall x \in I_h,\ D_{h,\alpha,\beta,q}^{(n)} f(x-th) = \frac{1}{(-h)^n} \frac{d^n}{dt^n} \left[D_{h,\alpha,\beta,q}^{(0)} f(x-th) \right]. \tag{7}$$

Moreover, we have

$$\forall x \in I_h,\ D_{h,\alpha,\beta,q}^{(n)} f(x-th) = \frac{1}{(-h)^n} \int_0^1 Q_{\alpha,\beta,n,q,t}(\tau) f(x-h\tau) d\tau, \tag{8}$$

where $Q_{\alpha,\beta,n,q,t}(\tau) = w_{\alpha,\beta}(\tau) \sum_{i=0}^{q} C_{\alpha,\beta,n,i} P_i^{(\alpha+n,\beta+n)}(t) P_{n+i}^{(\alpha,\beta)}(\tau)$, *and* $C_{\alpha,\beta,n,i} = \frac{(\beta+\kappa+2n+2i+1)\Gamma(\beta+\alpha+2n+i+1)\Gamma(n+i+1)}{\Gamma(\beta+n+i+1)\Gamma(\alpha+n+i+1)}$ *with* $\alpha, \beta \in]-1, +\infty[$.

Proof. By applying n times derivations to (6) and by using (4), we obtain

$$\begin{aligned}
&\frac{d^n}{dt^n}\left[D^{(0)}_{h,\alpha,\beta,q} f(x-th)\right] \\
=&\sum_{i=0}^{q} \frac{\left\langle P^{(\alpha,\beta)}_{i+n}(\cdot), f(x-h\cdot)\right\rangle_{\alpha,\beta}}{\|P^{(\alpha,\beta)}_{i+n}\|^2_{\alpha,\beta}} \frac{d^n}{dt^n}\left[P^{(\alpha,\beta)}_{i+n}(t)\right] \\
=&\sum_{i=0}^{q} \frac{\left\langle P^{(\alpha,\beta)}_{i+n}(\cdot), f(x-h\cdot)\right\rangle_{\alpha,\beta}}{\|P^{(\alpha,\beta)}_{i+n}\|^2_{\alpha,\beta}} \frac{\Gamma(\alpha+\beta+2n+i+1)}{\Gamma(\alpha+\beta+n+i+1)} P^{(\alpha+n,\beta+n)}_{i}(t).
\end{aligned} \tag{9}$$

By applying two times the Rodrigues formula given in (3) and by taking n integrations by parts, we get

$$\begin{aligned}
&\left\langle P^{(\alpha+n,\beta+n)}_{i}(\cdot), f^{(n)}(x-h\cdot)\right\rangle_{\alpha+n,\beta+n} \\
=&\int_0^1 w_{\alpha+n,\beta+n}(\tau) P^{(\alpha+n,\beta+n)}_{i}(\tau)\, f^{(n)}(x-h\tau)d\tau \\
=&\int_0^1 \frac{(-1)^i}{i!} w^{(i)}_{\alpha+n+i,\beta+n+i}(\tau)\, f^{(n)}(x-h\tau)d\tau \\
=&\frac{1}{(-h)^n}\int_0^1 \frac{(-1)^{i+n}}{i!} w^{(n+i)}_{\alpha+n+i,\beta+n+i}(\tau)\, f(x-h\tau)d\tau \\
=&\frac{1}{(-h)^n}\int_0^1 \frac{(n+i)!}{i!} w_{\alpha,\beta}(\tau) P^{(\alpha,\beta)}_{n+i}(\tau)\, f(x-h\tau)d\tau.
\end{aligned}$$

Then, after some calculations by using (2) we can obtain

$$\begin{aligned}
&\frac{\left\langle P^{(\alpha+n,\beta+n)}_{i}(\cdot), f^{(n)}(x-h\cdot)\right\rangle_{\alpha+n,\beta+n}}{\|P^{(\alpha+n,\beta+n)}_{i}\|^2_{\alpha+n,\beta+n}} \\
=&\frac{\left\langle P^{(\alpha,\beta)}_{i+n}(\cdot), f(x-h\cdot)\right\rangle_{\alpha,\beta}}{(-h)^n\|P^{(\alpha,\beta)}_{n+i}\|^2_{\alpha,\beta}} \frac{\Gamma(\alpha+\beta+2n+i+1)}{\Gamma(\alpha+\beta+n+i+1)}.
\end{aligned} \tag{10}$$

Finally, by taking (5) and (9) we obtain

$$\forall x \in I_h,\ D^{(n)}_{h,\alpha,\beta,q} f(x-th) = \frac{1}{(-h)^n}\frac{d^n}{dt^n}\left[D^{(0)}_{h,\alpha,\beta,q} f(x-th)\right]. \tag{11}$$

The proof is complete. □

If we consider the noisy function f^δ, then it is sufficient to replace $f(x-h\cdot)$ in (8) by $f^\delta(x-h\cdot)$. In [14], for a given value $t_\tau \in [0,1]$, $D^{(n)}_{h,\alpha,\beta,q} f^\delta(x-t_\tau h)$ (with $\alpha,\beta \in \mathbb{N}$ and $q \leq \alpha+n$) was proposed as a point-wise estimate of $f^{(n)}(x)$ by admitting a time-delay $t_\tau h$. We assume here that these values α and β belong to $]-1,+\infty[$. This is possible due to the definition of the Jacobi polynomials. Contrary to [14], we do not have constraints on the value of the truncation order q. Moreover, the function $Q_{\alpha,\beta,n,q,t}$ is easier to calculate than the one given in [12]. Thus, we can define the extended point-wise estimators as follows.

Definition 1. *Let $f^{\delta} = f + \varpi$ be a noisy function, where $f \in C^n(I)$ and ϖ be a bounded and integrable noise with a noise level δ. Then a family of causal Jacobi estimators of $f^{(n)}$ is defined as*

$$\forall x \in I_h,\ D^{(n)}_{h,\alpha,\beta,q} f^{\delta}(x - t_\tau h) = \frac{1}{(-h)^n} \int_0^1 Q_{\alpha,\beta,n,q,t_\tau}(u) f^{\delta}(x - hu)du, \tag{12}$$

where $\alpha, \beta \in]-1,+\infty[$, $q, n \in \mathbb{N}$ and t_τ is a fixed value on $[0,1]$.

Hence, the estimation error comes from two sources : the remainder terms in the Jacobi series expansion of $f^{(n)}(x - h\cdot)$ and the noise part. In the following proposition, we study these estimation errors.

Proposition 1. *Let f^{δ} be a noisy function where $f \in \mathcal{C}^{n+1+q}(I)$ and ϖ be a bounded and integrable noise with a noise level δ. Assume that there exists $M_{n+1+q} > 0$ such that for any $x \in I$, $\left|f^{(n+q+1)}(x)\right| \leq M_{n+1+q}$, then*

$$\left\|D^{(n)}_{h,\alpha,\beta,q} f^{\delta}(x - t_\tau h) - f^{(n)}(x - t_\tau h)\right\|_{\infty} \leq C_{q,t_\tau} h^{q+1} + E_{q,t_\tau} \frac{\delta}{h^n}, \tag{13}$$

where $C_{q,t_\tau} = M_{n+1+q}\left(\frac{1}{(n+1+q)!}\int_0^1 \left|u^{n+1+q} Q_{\alpha,\beta,n,q,t_\tau}(u)\right| du + \frac{t_\tau^{q+1}}{(q+1)!}\right)$ and $E_{q,t_\tau} = \int_0^1 \left|Q_{\alpha,\beta,n,q,t_\tau}(u)\right| du$. Moreover, if we choose $h = \left[\frac{nE_{q,t_\tau}}{(q+1)C_{q,t_\tau}}\delta\right]^{\frac{1}{n+q+1}}$, then we have

$$\left\|D^{(n)}_{h,\alpha,\beta,q} f^{\delta}(x - t_\tau h) - f^{(n)}(x - t_\tau h)\right\|_{\infty} = \mathcal{O}(\delta^{\frac{q+1}{n+1+q}}). \tag{14}$$

Proof. By taking the Taylor series expansion of f at x, we then have for any $x \in I_h$ that there exists $\xi \in]x-h, x[$ such that

$$f(x - t_\tau h) = f_{n+q}(x - t_\tau h) + \frac{(-h)^{n+1+q} t_\tau^{n+1+q}}{(n+1+q)!} f^{(n+1+q)}(\xi), \tag{15}$$

where $f_{n+q}(x - t_\tau h) = \sum_{j=0}^{n+q} \frac{(-h)^j t_\tau^j}{j!} f^{(j)}(x)$ is the $(n+q)^{th}$ order truncated Taylor series expansion of $f(x - t_\tau h)$. By using (8) with $f_{n+q}(x - h\cdot)$ we obtain

$$f^{(n)}_{n+q}(x - t_\tau h) = \frac{1}{(-h)^n} \int_0^1 Q_{\alpha,\beta,n,q,t_\tau}(\tau) f_{n+q}(x - h\tau) d\tau. \tag{16}$$

Thus, by using (12) and (16) we obtain

$$\begin{aligned} &D^{(n)}_{h,\alpha,\beta,q} f(x - t_\tau h) - f^{(n)}_{n+q}(x - t_\tau h) \\ =&\frac{(-h)^{q+1}}{(n+1+q)!} \int_0^1 Q_{\alpha,\beta,n,q,t_\tau}(\tau) \tau^{n+1+q} f^{(n+1+q)}(\xi) d\tau. \end{aligned}$$

Consequently, if for any $x \in I$ $\left|f^{(n+1+q)}(x)\right| \leq M_{n+1+q}$, then by taking the q^{th} order truncated Taylor series expansion of $f^{(n)}(x-t_\tau h)$

$$f^{(n)}(x-t_\tau h) = f^{(n)}_{n+q}(x-t_\tau h) + \frac{(-h)^{1+q} t_\tau^{1+q}}{(1+q)!} f^{(n+1+q)}(\hat{\xi}),$$

we have

$$\begin{aligned} &\left\| D^{(n)}_{h,\alpha,\beta,q} f(x-t_\tau h) - f^{(n)}(x-t_\tau h) \right\|_\infty \\ \leq &\left\| D^{(n)}_{h,\alpha,\beta,q} f(x-t_\tau h) - f^{(n)}_{n+q}(x-t_\tau h) \right\|_\infty + \left\| f^{(n)}_{n+q}(x-t_\tau h) - f^{(n)}(x-t_\tau h) \right\|_\infty \\ \leq & h^{q+1} M_{n+1+q} \left(\frac{1}{(n+1+q)!} \int_0^1 \left| \tau^{n+1+q} Q_{\alpha,\beta,n,q,t_\tau}(\tau) \right| d\tau + \frac{t_\tau^{q+1}}{(q+1)!} \right). \end{aligned} \tag{17}$$

Since

$$\begin{aligned} \left\| D^{(n)}_{h,\alpha,\beta,q} f^\delta(x-t_\tau h) - D^{(n)}_{h,\alpha,\beta,q} f(x-t_\tau h) \right\|_\infty &= \left\| D^{(n)}_{h,\alpha,\beta,q} \left[f^\delta(x-t_\tau h) - f(x-t_\tau h) \right] \right\|_\infty \\ &\leq \frac{\delta}{h^n} \int_0^1 \left| Q_{\alpha,\beta,n,q}(\tau) \right| d\tau, \end{aligned}$$

by using (17) we get

$$\begin{aligned} \left\| D^{(n)}_{h,\alpha,\beta,q} f^\delta(x-t_\tau h) - f^{(n)}(x-t_\tau h) \right\|_\infty &\leq \left\| D^{(n)}_{h,\alpha,\beta,q} f^\delta(x-t_\tau h) - D^{(n)}_{h,\alpha,\beta,q} f(x-t_\tau h) \right\|_\infty \\ &\quad + \left\| D^{(n)}_{h,\alpha,\beta,q} f(x-t_\tau h) - f^{(n)}(x-t_\tau h) \right\|_\infty \\ &\leq C_{q,t_\tau} h^{q+1} + E_{q,t_\tau} \frac{\delta}{h^n}, \end{aligned}$$

where $C_{q,t_\tau} = M_{n+1+q} \left(\frac{1}{(n+1+q)!} \int_0^1 \left| \tau^{n+1+q} Q_{\alpha,\beta,n,q,t_\tau}(\tau) \right| d\tau + \frac{t_\tau^{q+1}}{(q+1)!} \right)$ and $E_{q,t_\tau} = \int_0^1 \left| Q_{\alpha,\beta,n,q,t_\tau}(\tau) \right| d\tau$.

Let us denote the error bound by $\psi(h) = C_{q,t_\tau} h^{q+1} + E_{q,t_\tau} \frac{\delta}{h^n}$. Consequently, we can calculate its minimum value. It is obtained for $h^* = \left[\frac{n E_{q,t_\tau}}{(q+1) C_{q,t_\tau}} \delta \right]^{\frac{1}{n+q+1}}$ and

$$\psi(h^*) = \frac{n+1+q}{q+1} \left(\frac{q+1}{n} \right)^{\frac{n}{n+1+q}} C_{q,t_\tau}^{\frac{n}{n+1+q}} E_{q,t_\tau}^{\frac{q+1}{n+1+q}} \delta^{\frac{q+1}{n+1+q}}. \tag{18}$$

The proof is complete. □

Let us mention that if we set $t_\tau = \theta_{q+1}$, the smallest root of the Jacobi polynomial $P^{(\alpha+n,\beta+n)}_{q+1}$ in (5), then $D^{(n)}_{h,\alpha,\beta,q} f(x-\theta_{q+1}h)$ becomes the $(q+1)^{th}$ order truncated Jacobi orthogonal series of $f^{(n)}(x-\theta_{q+1}h)$. Hence, we have the following corollary.

Corollary 1. *Let $f \in \mathcal{C}^{n+2+q}(I)$ where q is an integer. If we set $t_\tau = \theta_{q+1}$, the smallest root of the Jacobi polynomial $P_{q+1}^{(\alpha+n,\beta+n)}$ in (12) and we assume that there exists $M_{n+2+q} > 0$ such that for any $x \in I$, $\left|f^{(n+q+2)}(x)\right| \leq M_{n+2+q}$, then we have*

$$\left\|D_{h,\alpha,\beta,q}^{(n)} f^\delta(x - \theta_{q+1}h) - f^{(n)}(x - \theta_{q+1}h)\right\|_\infty \leq C_{q,\theta_{q+1}} h^{q+2} + E_{q,\theta_{q+1}} \frac{\delta}{h^n},$$

where $C_{q,\theta_{q+1}} = M_{n+2+q}\left(\frac{1}{(n+q+2)!}\int_0^1 \left|\tau^{n+q+2} Q_{\alpha,\beta,n,q,\theta_{q+1}}(\tau)\right| d\tau + \frac{t^{q+2}}{(q+2)!}\right)$ and $E_{q,\theta_{q+1}}$ is given in Proposition 1. Moreover, if we choose $\hat{h} = \left[\frac{nE_{q,\theta_{q+1}}}{(q+2)C_{q,\theta_{q+1}}}\delta\right]^{\frac{1}{n+q+2}}$, then we have

$$\left\|D_{h,\alpha,\alpha,q}^{(n)} f^\delta(x - \theta_{q+1}h) - f^{(n)}(x - \theta_{q+1}h)\right\|_\infty = \mathcal{O}(\delta^{\frac{q+2}{n+2+q}}).$$

Proof. If $t_\tau = \theta_{q+1}$, the smallest root of polynomial $P_{q+1}^{(\alpha+n,\beta+n)}$ in (5), then we have

$$\begin{aligned} &D_{h,\alpha,\beta,q}^{(n)} f(x - \theta_{q+1}h) \\ &\quad = \sum_{i=0}^{q+1} \frac{\left\langle P_i^{(\alpha+n,\beta+n)}(\cdot), f^{(n)}(x - h\cdot)\right\rangle_{\alpha+n,\beta+n}}{\|P_i^{(\alpha+n,\beta+n)}\|_{\alpha+n,\beta+n}^2} P_i^{(\alpha+n,\beta+n)}(\theta_{q+1}). \end{aligned}$$

This proof can be completed by taking the $(n+q+1)^{th}$ order truncated Taylor series expansion of f as it was done in Proposition 1. □

The numerical calculation of $E_{q,\theta_{q+1}}$ for q, $n \in \mathbb{N}$ and $\alpha, \beta \in]-1, 10]$ shows that $E_{q,\theta_{q+1}}$ increases with respect to q. Hence, in order to reduce the noise influence it is preferable to choose q as small as possible. However, $C_{q,\theta_{q+1}}$ decreases with respect to q. A compromise consists in choosing $q = 2$. If we take $D_{h,\alpha,\beta,2}^{(n)} f^\delta(x - \theta_2 h)$ as an estimator of $f^{(n)}(x)$, then we produce a time-delay $\theta_2 h$. For this choice of θ_2, we have $D_{h,\alpha,\beta,1}^{(n)} f^\delta(x - \theta_2 h) = D_{h,\alpha,\beta,2}^{(n)} f^\delta(x - \theta_2 h)$. We can see that the estimators $D_{h,\alpha,\beta,2}^{(n)} f^\delta(x)$ do not produce a time-delay but $C_{1,\theta_2} < C_{2,0}$ and $E_{1,\theta_2} < E_{2,0}$. This generally introduces more important estimation errors. Consequently, so as to estimate $f^{(n)}(x)$ we use $D_{h,\alpha,\beta,1}^{(n)} f^\delta(x - \theta_2 h)$ which presents a time-delay.

3 Numerical Experiments

In order to show the efficiency and the stability of the previously proposed estimators, we give some numerical results in this section.

From now on, we assume that $f^\delta(x_i) = f(x_i) + c\varpi(x_i)$ with $x_i = T_s i$ for $i = 0, \cdots, 500$ ($T_s = \frac{1}{100}$), is a noisy measurement of $f(x) = \exp(\frac{-x}{1.2})\sin(6x + \pi)$. The noise $c\varpi(x_i)$ is simulated from a zero-mean white Gaussian *iid* sequence by using the Matlab function 'randn' with STATE reset to 0. Coefficient c is adjusted in such a way that the signal-to-noise ratio $SNR = 10\log_{10}\left(\frac{\sum |y(t_i)|^2}{\sum |c\varpi(t_i)|^2}\right)$ is equal to $SNR = 22.2$dB (see, e.g., [7] for this well known concept in signal processing). By using the well known three-sigma rule, we can assume that the noise level for $c\varpi$ is equal to $3c$. We can see the noisy signal in Figure 1. We use the trapezoidal method in order to approximate the integrals in our estimators where we use $m+1$ discrete values. The estimated derivatives of f at $x_i \in I = [0, 5]$ are calculated from the noise data $f^\delta(x_j)$ with $x_j \in [-x_i - h, x_i]$ where $h = mT_s$.

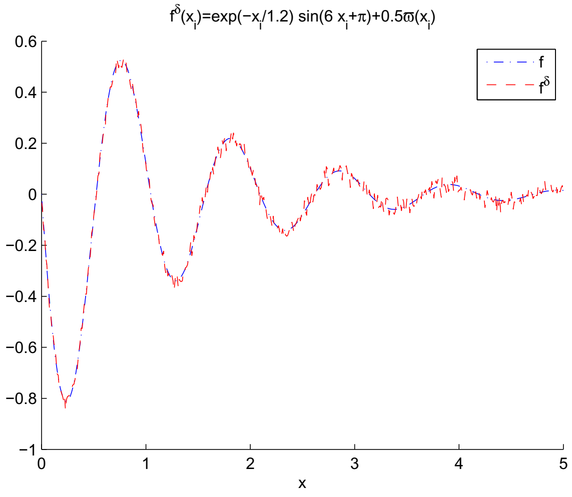


Fig. 1. Signal and noisy signal

We can see the estimation results for the first order derivative of f in Figure 2. The corresponding estimation errors are given in Figure 3 and in Figure 4. We can see that the estimate given by the causal Jacobi estimator with integer parameters introduced in [14] (dash line), produces a time-delay of value $\theta_{q+1}h = 0.11$. The estimate given by the causal Jacobi estimator with extended parameters (dotted line) is time-delay free. Firstly, the root values θ_{q+1} for $q = 0, 1$ can be reduced with the extended negative parameters, so does the time-delay. Secondly, the numerical integration method with a negative value for β produces a numerical error which allows us to finally compensate this reduced time-delay. This last phenomena is due to the fact that the initial function f satisfies the following differential equation $f^{(2)} + kf = g$ where $k \in \mathbb{R}$ and g is a continuous function. Consequently, in the case of the first order derivative estimations, we can verify that this numerical error which depends on f may reduce the effect of the error due to the truncation in the Jacobi series expansion. This is due to the fact that the truncation error depends on $f^{(2)}$. Hence, the final total error is $\mathcal{O}(g)$. Finally, since $D^{(1)}_{mT_s,\alpha,\beta,q} f^\delta(x_i - \theta_2 h)$ produces a time-delay of value $\theta_2 h$, we give in Figure 5 the errors $D^{(1)}_{mT_s,\alpha,\beta,q} f^\delta(x_i - \theta_2 h) - f^{(1)}(x_i - \theta_2 h)$.

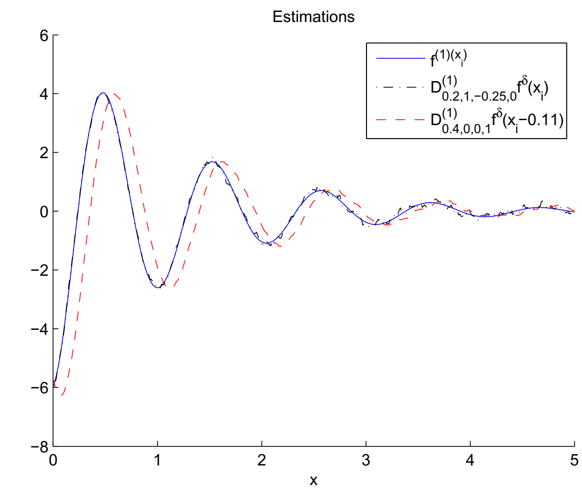


Fig. 2. Estimations by Jacobi estimators $D^{(1)}_{h,\alpha,\beta,q}f^{\delta}(x_i)$ with $h = 0.2, \beta = -0.25, \alpha = 1, q = 0$ and $D^{(1)}_{h,\alpha,\beta,q}f^{\delta}(x_i - \theta_2 h)$ with $h = 0.4, \alpha = \beta = 0, q = 1, \theta_2 = 0.276$

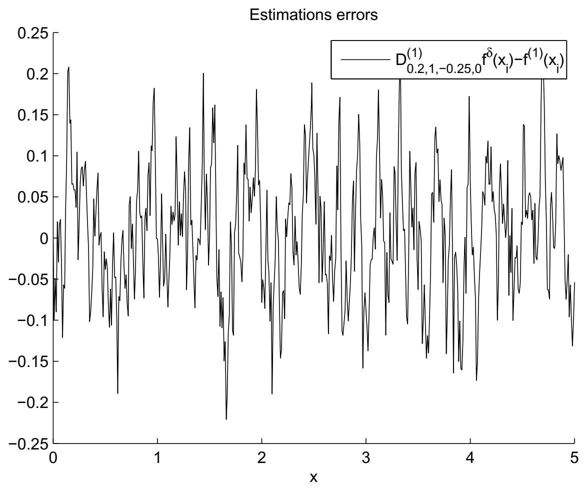


Fig. 3. $D^{(1)}_{h,\alpha,\beta,q}f^{\delta}(x_i) - f^{(1)}(x_i)$ with $h = 0.2, \beta = -0.25, \alpha = 1, q = 0$

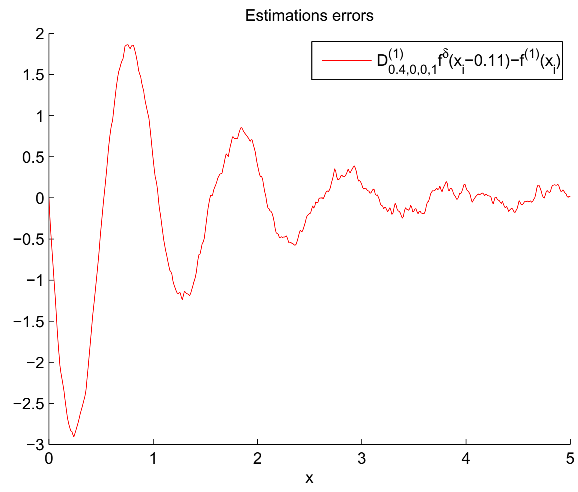


Fig. 4. $D^{(1)}_{h,\alpha,\beta,q} f^{\delta}(x_i - \theta_2 h) - f^{(1)}(x_i)$ with $h = 0.4, \alpha = \beta = 0, q = 1, \theta_2 = 0.276$

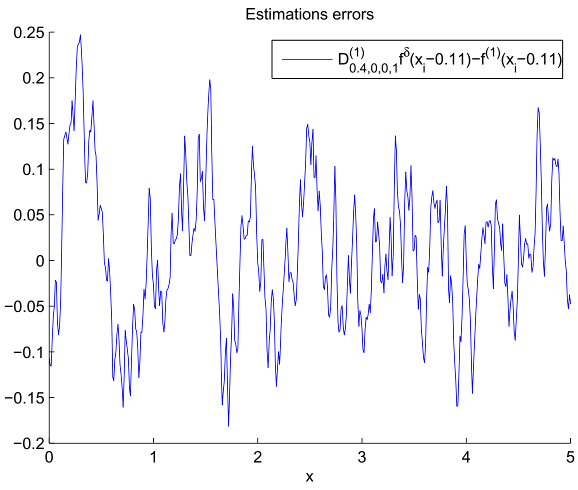


Fig. 5. $D^{(1)}_{h,\alpha,\beta,q} f^{\delta}(x_i - \theta_2 h) - f^{(1)}(x_i - \theta_2 h)$ with $h = 0.4, \alpha = \beta = 0, q = 1, \theta_2 = 0.276$

References

1. Cheng, J., Hon, Y.C., Wang, Y.B.: A numerical method for the discontinuous solutions of Abel integral equations. In: Inverse Problems and Spectral Theory, Contemp., vol. 348, pp. 233–243. Amer. Math. Soc., Providence (2004)
2. Fliess, M., Sira-Ramírez, H.: An algebraic framework for linear identification. ESAIM Control Optim. Calc. Variat. 9, 151–168 (2003)

3. Fliess, M., Join, C., Mboup, M., Sira-Ramírez, H.: Compression différentielle de transitoires bruités. C.R. Acad. Sci. I(339), 821–826 (2004)
4. Fliess, M.: Analyse non standard du bruit. C.R. Acad. Sci. Paris Ser. I. 342, 797–802 (2006)
5. Fliess, M.: Critique du rapport signal à bruit en communications numériques – Questioning the signal to noise ratio in digital communications. In: International Conference in Honor of Claude Lobry, ARIMA (Revue africaine d'informatique et de Mathématiques appliquées), vol. 9, pp. 419–429 (2008), http://hal.inria.fr/inria-00311719/en/
6. Fliess, M., Mboup, M., Mounier, H., Sira-Ramírez, H.: Questioning some paradigms of signal processing via concrete examples. In: Sira-Ramírez, H., Silva-Navarro, G. (eds.) Algebraic Methods in Flatness, Signal Processing and State Estimation, Editiorial Lagares, México, pp. 1–21 (2003)
7. Haykin, S., Van Veen, B.: Signals and Systems, 2nd edn. John Wiley & Sons, Chichester (2002)
8. Ibrir, S.: Linear time-derivatives trackers. Automatica 40, 397–405 (2004)
9. Lanczos, C.: Applied Analysis. Prentice-Hall, Englewood Cliffs (1956)
10. Liu, D.Y., Gibaru, O., Perruquetti, W., Fliess, M., Mboup, M.: An error analysis in the algebraic estimation of a noisy sinusoidal signal. In: 16th Mediterranean Conference on Control and Automation (MED 2008), Ajaccio (2008)
11. Liu, D.Y., Gibaru, O., Perruquetti, W.: Error analysis of Jacobi derivative estimators for noisy signals. Numerical Algorithms, doi:10.1007/s11075-011-9447-8
12. Liu, D.Y., Gibaru, O., Perruquetti, W.: Differentiation by integration with Jacobi polynomials. Journal of Computational and Applied Mathematics 235, 3015–3032 (2011)
13. Mboup, M.: Parameter estimation for signals described by differential equations. Applicable Analysis 88, 29–52 (2009)
14. Mboup, M., Join, C., Fliess, M.: Numerical differentiation with annihilators in noisy environment. Numerical Algorithms 50(4), 439–467 (2009)
15. Mboup, M., Join, C., Fliess, M.: A revised look at numerical differentiation with an application to nonlinear feedback control. In: 15th Mediterranean Conference on Control and Automation (MED 2007), Athenes, Greece (2007)
16. Murio, D.A., Mejía, C.E., Zhan, S.: Discrete mollification and automatic numerical differentiation. Comput. Math. Appl. 35, 1–16 (1998)
17. Nakamura, G., Wang, S., Wang, Y.: Numerical differentiation for the second order derivatives of functions of two variables. Journal of Computational and Applied Mathematics 212, 341–358 (2008)
18. Rader, C.M., Jackson, L.B.: Approximating noncausal IIR digital filters having arbitrary poles, including new Hilbert transformer designs, via forward/backward block recursion. IEEE Trans. Circuits Syst. I 53(12), 2779–2787 (2006)
19. Rangarajana, S.K., Purushothaman, S.P.: Lanczos' generalized derivative for higher orders. Journal of Computational and Applied Mathematics 177, 461–465 (2005)
20. Ramm, A.G., Smirnova, A.B.: On stable numerical differentiation. Math. Comput. 70, 1131–1153 (2001)
21. Szegö, G.: Orthogonal polynomials, 3rd edn. AMS, Providence (1967)
22. Wang, Z., Liu, J.: Identification of the pollution source from one-dimensional parabolic equation models. Applied Mathematics and Computation (2008), doi:10.1016/j.amc.2008.03.014
23. Wang, Z., Wen, R.: Numerical differentiation for high orders by an integration method. Journal of Computational and Applied Mathematics 234, 941–948 (2010)

Continuous Deformations by Isometry Preserving Shape Integration

Janick Martinez Esturo, Christian Rössl, and Holger Theisel

Visual Computing Group, University of Magdeburg, Germany

Abstract. We introduce a novel continuous surface deformation method which relies on a time-dependent vector field over a triangular mesh. For every time step the piecewise linear vector field is obtained by least-squares minimization of the metric distortion induced by integration subject to boundary conditions. As an integral part of the approach, we introduce a new measure to describe local metric distortion which is invariant to the particular triangulation of the surface and which can incorporate smoothness of the field. Neither of these properties are met by previous work. A GPU implementation of the proposed algorithm enables fast deformations. The resulting deformations have lower metric distortions than deformations by existing (linear or non-linear) methods. This is shown for a number of representative test data sets.

Keywords: Shape Deformation, Isometry, Vector Field.

1 Introduction

Shape deformations constitute a standard problem in modeling and computer graphics. A variety of approaches have been proposed in the recent decade, and shape deformation is still an active area of research. A good approach to deformations should be intuitive, visually convincing, geometrically or/and physically sound, and reasonably fast.

We consider shapes represented as triangle meshes. Most current approaches define deformations as a minimization problem. Given certain boundary conditions, the unknown vertex positions have to be determined. This yields a linear or non-linear optimization problem depending on the measure to be minimized. Proceeding this way, only the final positions of the vertices are computed without considering the deformation path leading to the final state. We call this kind of deformations *discrete deformations*. In contrast, *continuous deformations* integrate vertex positions along smooth vector fields. Here, the boundary conditions of the deformation are paths of parts of the shape. In this sense, we are aware of only two approaches to continuous deformations so far: [14] and its extension [15] which applies divergence-free vector fields for volume preserving shape deformation, and [18] which considers the deformations in a shape space.

Various measures have been proposed for minimization in discrete shape deformations. These measures typically reflect or simulate physical properties like

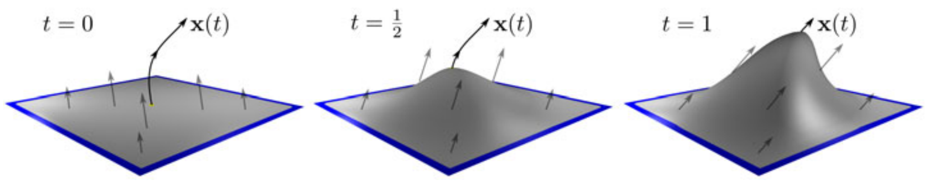


Fig. 1. Optimization Problem at three Time Steps of the Integration. The small handle region is marked yellow while the blue border corresponds to the fixed surface part. For every integration step, the current vertex positions and black vectors are given, while the gray vectors are obtained by solving a sparse linear system

bending energies which may be simplified or linearized. Only recently, minimization of metric distortion has been considered [18,17].

In this paper we present a new approach to continuous deformations which minimizes distortion of the surface and tries to preserve isometry. We define the deformation continuously over time, i.e., as a vector field which is determined in every time step as the result of a linear problem. The arising system matrices are sparse and can be solved reliably and efficiently supported by the GPU, which renders our approach reasonably fast. Figure 1 illustrates the main idea of our approach: following the standard deformation metaphor [4,24], we define regions of full and zero deformation of the shape. For regions of full deformation, the deformation is given as parametric curves. Then in every time step a piecewise linear vector field is constructed such that metric distortion is minimized under integration. Boundary conditions are defined by the tangent vectors of the parametric curves describing the full deformation.

In summary, the main contributions of this work are: (1) design of a new discrete isometry measure which is invariant to surface tessellation and which extends naturally to incorporate smoothness. (2) Definition of continuous, isometry preserving shape deformations which are constrained by trajectories of surface regions. The whole approach is completely geometry-driven.

The remainder of the paper is organized as follows. Section 2 reviews related work. Section 3 introduces our approach and defines all measures while Section 4 discusses how the theoretic concepts are applied and implemented. Results are presented in Section 5 followed by a discussion of the method in Section 6 and final conclusions (Section 7).

2 Related Work

There is a vast amount of literature on shape deformation, a proper review is far beyond the scope of this section. Instead, we point to the recent survey of Botsch and Sorkine [6]: they discuss and compare the most important classes of explicit surface deformation methods. In fact, we use their reference deformations as benchmarks. All reviewed methods implement discrete deformations which are completely determined by boundary constraints, i.e., the placement of surface regions, fixed or handle regions, in 3-space. Deformation of the surface is then modeled in one of several ways:

- as a variational problem minimizing an energy functional which penalizes certain bending energies (see, e.g., [4]),
- as reconstruction from any kind of differential coordinates (see, e.g., [19]),
- as a projection or Poisson reconstruction after application of a "transformation field" to individual triangles thus over-determining vertex positions (see, e.g., [27]), or
- as simulation of forces to rigid and loosely coupled prism elements enveloping the surface [5].

Several methods are closely related, and they all share the goals of feature preservation and establishing smooth transitions towards deformed regions. All of these methods (except [5]) rely on the factorization of few or even only one single linear systems, a fact that renders these methods interactive. In particular, movement of handles requires only back-substitution for solving the system.

Furthermore, there is a variety of methods which determine a piecewise deformation where individual pieces are close to rigid transformations, i.e, they ought to be as rigid as possible. We refer to recent work by Sumner et al. [25] (and the references therein): here, a free-form deformation is determined based on a user provided "deformation graph", which defines the piecewise deformation. A non-linear minimization determines the degrees of freedom for the individual transformation pieces associated with the nodes. We note that our goal is not to obtain an as-rigid-as possible deformation but to stay as-isometric-as possible, and we refer to [18] for a more elaborate discussion. Furthermore our approach does not define a space warp but an explicit surface deformation (evaluation of our guiding vector fields is meaningful only for surface points), and we explicitly include the surface metric. Of course this review cannot be complete as there is a vast amount of literature on shape deformation. Among other non-linear methods we mention [3,16,26] as recent examples.

For all the above mentioned methods, deformation remains a *discrete* process: it is solely the rest position of the handles that determines the result but not their particular trajectory.

In contrast, von Funck et al. [14] introduce an approach based on integration of a surface along a time-dependent vector field. Their goal is volume preserving deformations. This is achieved in an elegant way: the guiding vector fields are constructed to show zero divergence. Remarkable in the context of our work is that this constitutes a *continuous* deformation which does depend on the particular trajectories of the handles. (Note that otherwise this method is fundamentally different to ours.) The method was extended in [15] by the introduction of deformation paths. An earlier approach [2] related to this does not rely on continuous vector fields but instead discretizes the deformation.

Kilian et al. [18] regard continuous deformations in a shape space: they solve a boundary value problem in order to find a time-dependent deformation

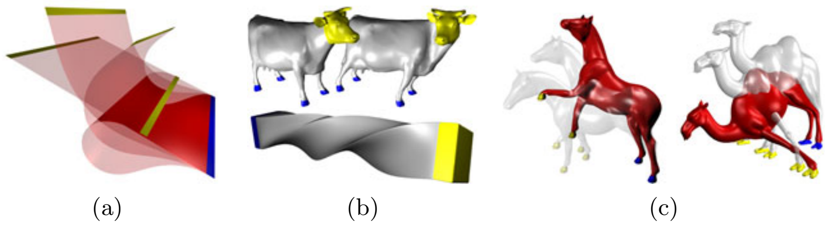


Fig. 2. Deformation examples. (a) Perfectly isometric deformation of developable plane. (b) Twisting deformation of the head of the cow and a strong twist of a bar by 280° which is not achievable by discrete deformations. (c) Continuous deformations using multiple handles.

between two poses of the same shape. They advocate to determine optimal deformations by isometry preservation rather than a more traditional as-rigid-as possible [1] criterion. Then a discrete version of Killing fields [8] characterizes time-dependent deformation vector fields. While the focus is on boundary value problems and their solution by a space-time multigrid approach, initial value problems are discussed briefly. In both cases, the approach is not interactive.

We consider the latter deformation approaches by Funck et al. [14] and Kilian et al. [18] as *continuous* deformation methods. Discrete methods mentioned above rely on the minimization of certain (potentially linearized) energy functionals which often results in solving associated Euler-Lagrange equations characterizing an equilibrium state. While this is clearly different from our setting, it is also obvious that such discrete methods which are not modeled by a time-dependent flow can easily be modified such that a single deformation is broken into multiple steps defined, e.g., along a path (Section 6 shows an experiment). In fact, this may be regarded as a simple approach to emulating parametrization-independent, non-linear operators [6].

Isometry preserving deformations have been studied extensively in differential geometry (see, e.g., [8]) and mathematics in general: Efimow [12] theoretically investigated infinitesimal first order and higher order deformations. In this work we consider the first order case for piecewise linear discrete surfaces.

Finally, we remark that there are scenarios where physical objects including thin shells or cloth are modeled and shapes (or solids) obtain measured or synthesized material attributes. In a series of papers inspired by physically-based settings, Qin et al.[22,10] consider time-dependent surfaces using a dynamic FEM formulation for shape modeling. Emphasis is on the design of smooth surfaces, the process is governed by a mass-spring system. For recent work on cloth simulation in the context of developable surfaces we refer to [13]. Generally such deformations must be physically correct or at least plausible. While this is far from our goals we remark that such simulations may be seen as continuous processes, see, e.g., the survey of Nealen et al.[20].

3 Minimum Distortion Shape Integration

Our method is based on penalizing metric distortion when integrating vertex positions along a piecewise linear vector field. In this section we describe our approach in detail. We start with the derivation of the error measure for a single triangle. Contributions of each triangle are accumulated on the entire shape. Finally, we show how the concept can be extended naturally to incorporate smoothness of the resulting vector fields.

3.1 Error Measure on a Single Triangle

Reviewing the situation for a single triangle T is sufficient to explain the error measure. We consider a triangle with vertex positions $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$ and associated vectors $\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^3$. The triangle surface $\mathbf{x}$ and the vector field $\mathbf{v}$ are obtained by linear interpolation as

$$\begin{aligned}\mathbf{x}_T(u,v) &= u\,\mathbf{x}_0 \;+\; v\,\mathbf{x}_1 \;+\; (1-u-v)\,\mathbf{x}_2\\ \mathbf{v}_T(u,v) &= u\,\mathbf{v}_0 \;+\; v\,\mathbf{v}_1 \;+\; (1-u-v)\,\mathbf{v}_2 .\end{aligned}$$

The subsequent derivation is motivated by the fact that integrating perfectly rigid vector fields yields zero distortion. Obviously they won't yield a reasonable deformation either. However, local rigid fields can be easily constructed and serve as a reference: the closer $\mathbf{v}_T$ is to a rigid field, the more isometric the deformation.

We consider a 3D vector field $\mathbf{r}(\mathbf{x})$ describing a *rigid* vector field, i.e., it can be written as $\mathbf{r}(\mathbf{x}) = \mathbf{r}_t + (\mathbf{r}_r \times \mathbf{x})$. Here, $\mathbf{r}_t$ and $\mathbf{r}_r$ describe the translational part and the rotation axis, respectively. Note that even though $\mathbf{r}$ is defined everywhere in $\mathbb{R}^3$, we evaluate it only on the triangle. We define the error e_T as squared difference of $\mathbf{r}$ and $\mathbf{v}$ integrated over the triangle T:

$$e_T(\mathbf{x},\mathbf{v},\mathbf{r}) = \int_0^1 \int_0^{1-v} ||\,\mathbf{v}(u,v) - \mathbf{r}(\mathbf{x}(u,v))\,||^2 \,du\,dv \; . \tag{1}$$

This can be expressed in closed form as

$$e_T(\mathbf{x},\mathbf{v},\mathbf{r}) = \frac{1}{6} \sum_{(i,j)\in\{(0,1),(1,2),(2,0)\}} ||\mathbf{v}_{ij} - \mathbf{r}(\mathbf{x}_{ij})||^2$$

with $\mathbf{x}_{ij} = \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_j)$ and $\mathbf{v}_{ij} = \frac{1}{2}(\mathbf{v}_i + \mathbf{v}_j)$.

Given $\mathbf{x}$, our goal is to compute the best fitting rigid vector field $\hat{\mathbf{r}}(\mathbf{x},\mathbf{v})$ as a function of $\mathbf{v}$ by minimizing $e_T(\mathbf{x},\mathbf{v},\mathbf{r})$ for all rigid fields $\mathbf{r}$:

$$\hat{\mathbf{r}}(\mathbf{x},\mathbf{v}) = \operatorname*{argmin}_{\mathbf{r}} e_T(\mathbf{x},\mathbf{v},\mathbf{r}) \; . \tag{2}$$

This is a linear least-squares problem in the six coefficients of $\mathbf{r}$ (see also Section 4.2). Its solution depends on both, the positions $\mathbf{x}_i$ and vectors $\mathbf{v}_i, i = 0, 1, 2$.

Finally, we express metric distortion of $\mathbf{x}$ under instantaneous motion along $\mathbf{v}$ as

$$d_T(\mathbf{x},\mathbf{v}) = e_T(\mathbf{x},\mathbf{v},\hat{\mathbf{r}}(\mathbf{x},\mathbf{v})) \; . \tag{3}$$

3.2 Properties of d

The measure d_T is invariant under adding a rigid field to $\mathbf{v}$: let $\hat{\mathbf{r}}$ be the best fitting rigid field for $\mathbf{x}$ and $\mathbf{v}$, and let $\mathbf{p}(\mathbf{x})$ be another arbitrary rigid vector field. Then the best fitting rigid field for $\mathbf{x}$ and the modified vectors $\mathbf{v}'_i = (\mathbf{v}_i + \mathbf{p}(\mathbf{x}_i)), i = 0, 1, 2$, is $\hat{\mathbf{r}} + \mathbf{p}$. Furthermore, $d_T(\mathbf{x}, \mathbf{v}) = d_T(\mathbf{x}, \mathbf{v}')$.

The computation of $\hat{\mathbf{r}}$ is robust as long as the triangle is not degenerated, i.e., as long as the triangle area and the edge length ratios are bounded from below.

We emphasize that by construction $d_T(\mathbf{x}, \mathbf{v})$ measures metric distortion: isometric distortions of developable surfaces (see Figure 2a) indeed yield zero distortion although the deformation is not rigid. In particular this differs from as-rigid-as possible formulations (see [18] for a comparative discussion).

In the literature [18,11], (discrete) metric distortion of a triangle under integration of its vertices is usually described as

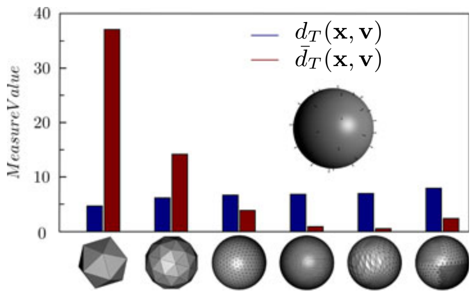
$$\bar{d}_T(\mathbf{x}, \mathbf{v}) = h_0^2 + h_1^2 + h_2^2 \quad \text{with} \quad h_k = \mathbf{r}_k^\top\,(\mathbf{v}_{\pi_{k+1}} - \mathbf{v}_{\pi_k}),\ \mathbf{r}_k = (\mathbf{x}_{\pi_{k+1}} - \mathbf{x}_{\pi_k}), \quad (4)$$

and $\pi_k = (k+1) \mod 3$. Then the summation of $\bar{d}_T$ over all triangles is the global measure to be minimized. Our measure is related and compatible to this in the sense that $d_T(\mathbf{x}, \mathbf{v}) = (h_0, h_1, h_2)\,\mathbf{S}\,(h_0, h_1, h_2)^\top$. Let $r_{ij} = \mathbf{r}_i^\top\,\mathbf{r}_j$ and $\alpha = 4\ \mathrm{area}(T)^2$. Then $\mathbf{S}$ is a symmetric 3×3 matrix which depends only on $\mathbf{x}$ and not on $\mathbf{v}$:

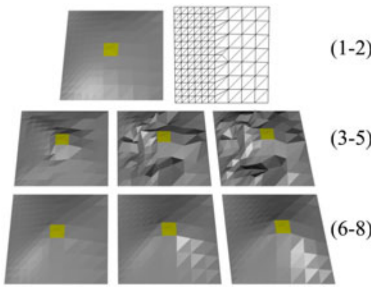
$$\mathbf{S} = \frac{-1}{144\,\alpha\,(r_{12} + r_{20} + r_{01})} \begin{bmatrix} 3\,r_{12}^2 + 4\,\alpha & 6\,r_{12}\,r_{20} - 4\,\alpha & 6\,r_{20}\,r_{01} - 4\,\alpha \\ 6\,r_{20}\,r_{12} - 4\,\alpha & 3\,r_{20}^2 + 4\,\alpha & 6\,r_{01}\,r_{12} - 4\,\alpha \\ 6\,r_{01}\,r_{20} - 4\,\alpha & 6\,r_{12}\,r_{01} - 4\,\alpha & 3\,r_{01}^2 + 4\,\alpha \end{bmatrix}.$$

Hence, generally no edge or area weighting–scheme exists which turns $\bar{d}_T(\mathbf{x}, \mathbf{v})$ into $d_T(\mathbf{x}, \mathbf{v})$, since $\mathbf{S}$ is generally not diagonal. In contrast to $\bar{d}_T(\mathbf{x}, \mathbf{v})$ and roughly speaking, our measure $d_T(\mathbf{x}, \mathbf{v})$ also incorporate all mixed products $h_i h_j, i \neq j$.

The integration of quantities over triangles is essential to the design of our measure: this way, $d_T(\mathbf{x}, \mathbf{v})$ will be *invariant* to subdivision of triangles or generally independent of parametrization/tessellation. Figure 3a illustrates this by a simple example and compares $d_T(\mathbf{x}, \mathbf{v})$ to $\bar{d}_T(\mathbf{x}, \mathbf{v})$: we prescribe a vector field and apply the measure for different tessellations of the same shape, a unit sphere. Then tessellation-independence requires low *variance* of measured values. Geometrically the absolute values are meaningless for this experiment. However, they can physically be interpreted to be the applied membrane strain since isometric deformations are a geometric approximation of real-life, thin surfaces deforming with a very high Young's modulus [20]. Further experiments and a comparison are shown in Figures 6 and 3b (see Section 6). Tessellation-independence is generally an important requirement for many algorithms. It is essential for meaningful deformations also because coherence for time–dependent deformations is improved.



(a) Tesselation Dependence



(b) Measure Stability

Fig. 3. Measure Comparison. (a) Different unit sphere tesselations and values of measures $d_T(\mathbf{x},\mathbf{v})$ and $\bar{d}_T(\mathbf{x},\mathbf{v})$ for the normal field. Important is the *variance* of the measure for different tessellations of the same shape which should be low. (b) The irregularly tessellated test surface (1–2) is Euler-integrated three steps by minimizing $\bar{d}_T(\mathbf{x},\mathbf{v})$ (3–5), and by minimizing $d_T(\mathbf{x},\mathbf{v})$ (6–8).

3.3 Error Measure on the Entire Surface

The shape $\mathcal{S}$ is given as a triangle mesh $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ with vertices $\mathcal{V}$, directed edges $\mathcal{E}$, and triangles $\mathcal{T}$. The embedding of the surface $\mathbf{x}$ in 3-space is defined by vertex positions $\mathbf{x}_i \in \mathbb{R}^3$, furthermore the piecewise linear vector field $\mathbf{v}$ is defined by vectors $\mathbf{v}_i, i = 1, \ldots, |\mathcal{V}|$.

We define the global metric distortion as

$$d_1(\mathbf{x},\mathbf{v}) = \sum_{(i,j,k)\in\mathcal{T}} d_T([\mathbf{x}_i,\mathbf{x}_j,\mathbf{x}_k],[\mathbf{v}_i,\mathbf{v}_j,\mathbf{v}_k])\,. \tag{5}$$

With (3) it is evident that $\mathbf{v}$ is a Killing field [8,12,18] of $\mathbf{x}$ iff $d_1(\mathbf{x},\mathbf{v}) = 0$. This also corresponds to intuition: if $\mathbf{v}$ is a Killing field, the best fitting rigid field for every triangle will be identical to $\mathbf{v}$ at each triangle, yielding $d_1(\mathbf{x},\mathbf{v}) = 0$.

An error measure which is based solely on the preservation of isometry is obviously not sufficient to determine meaningful surface deformation: for instance, foldings yield perfect isometry whereas for shape deformation they are considered unwanted artifacts. The measure d_1 does not exclude, e.g., foldings of a developable surface (see, e.g., [17]). Consequently, we require an additional measure which penalizes discontinuous deformation and enforces smoothness of the vector field $\mathbf{v}$.

A suitable measure that fits our setting should be derived from existing quantities. We take advantage of the fact that the best fitting rigid vector fields $\hat{\mathbf{r}}$ are defined not only on respective triangles but everywhere in $\mathbb{R}^3$. In particular, we can evaluate $\hat{\mathbf{r}}$ for a certain triangle T on an adjacent triangle T'.

Let triangles $T = (r,s,t)$ and $T' = (s,r,t')$ be adjacent with vertex positions $\mathbf{x}_\ell$, and associated vectors $\mathbf{v}_\ell$, $\ell \in \{r,s,t,t'\}$. Furthermore, let $\hat{\mathbf{r}}$ and $\hat{\mathbf{r}}'$ be the best fitting rigid fields on T and T'. Then, loosely spoken, $\hat{\mathbf{r}}$ and $\mathbf{v}_{T'}$ should not differ too much for a meaningful deformation. We formalize this by applying $\hat{\mathbf{r}}$

to T' (and $\hat{\mathbf{r}}'$ to T respectively). We obtain

$$f_{T,T'}(\mathbf{x},\mathbf{v}) = \int_0^1 \int_0^{1-v} ||\, \mathbf{v}_{T'}(u,v) - \hat{\mathbf{r}}(\mathbf{x}_{T'}(u,v))\, ||^2 \, du \, dv$$

where $\mathbf{x}_{T'}(u,v)$ and $\mathbf{v}_{T'}(u,v)$ denote the linear interpolation as before but on triangle T' whereas $\hat{\mathbf{r}}$ is the best fitting rigid field on T. This can be expressed in closed form

$$f_{T,T'}(\mathbf{x},\mathbf{v}) = \frac{1}{6} \sum_{(m,n)\in\{(r,s),(s,t'),(t',r)\}} ||\mathbf{v}_{mn} - \hat{\mathbf{r}}(\mathbf{x}_{mn})||^2$$

with $\mathbf{x}_{mn} = \frac{1}{2}(\mathbf{x}_m + \mathbf{x}_n)$ and $\mathbf{v}_{mn} = \frac{1}{2}(\mathbf{v}_m + \mathbf{v}_n)$. With the measure $f_{T,T'}$ on single triangles we define the global measure d_2 on the entire shape as

$$d_2(\mathbf{x},\mathbf{v}) = \sum_{T,T'\in\mathcal{T} \text{ adjacent}} f_{T,T'}(\mathbf{x},\mathbf{v}) \ .$$

In general $f_{T,T'}(\mathbf{x},\mathbf{v}) \neq f_{T',T}(\mathbf{x},\mathbf{v})$ which both have to be added in d_2. Botsch et al. [5] use a similar principle to match transformations of incident prisms.

With the derivation of d_1 and d_2 we finally define the error measure which will be minimized in our approach as

$$d(\mathbf{x},\mathbf{v}) = (1-\omega)\, d_1(\mathbf{x},\mathbf{v}) + \omega\, d_2(\mathbf{x},\mathbf{v}) \tag{6}$$

for a small weight $\omega > 0$ (see below).

We motivated the second measure d_2 by the fact that isometry does not always convey enough information for meaningful deformations. We remark that d_2 is required for another reason: in special cases, minimizing d_1 yields a singular or ill-conditioned operator, and d_2 acts as a regularization term. For instance, a planar surface constitutes this special case. Of course, then it applies only to the first integration step – after that the surface is probably no longer planar. However, to ensure robustness of our approach in any possible situation we require $\omega > 0$.

Our experiments show that a rather high value in $(0,1]$ can be chosen for ω without spoiling minimization of distortion, i.e., the effect of d_1. The reason for this is that the definition of d_2 contains essentials of d_1 just with the difference of extrapolating the rigid fields. We close this section with two final remarks: First, the errors d_1 and d_2 are compatible in a sense that comparable quantities are measured. Second, there is a bias in the weighting as summation is over $|\mathcal{T}|$ triangles for d_1 and over $|\mathcal{E}|$ adjacent triangles for d_2. From the latter relation Euler's formula yields $\omega = \frac{1}{3}$ for an even weighting.

3.4 Shape Integration

Given the shape with vertex positions $\mathbf{x}$, our goal is to find a piecewise linear vector field $\hat{\mathbf{v}}$ that minimizes the error functional $d(\mathbf{x},\mathbf{v})$. This vector field minimizes metric distortion under integration of $\mathbf{x}$ due to the definition of d_1. Additionally, the contribution of d_2 to d accounts for smoothness of $\hat{\mathbf{v}}$.

Our approach to isometry preserving shape deformation assumes time-dependent shape $\mathbf{x}(t)$ and vector field $\hat{\mathbf{v}}(\mathbf{x}, t)$ such that

$$\frac{\partial}{\partial t}\mathbf{x}(t) \;=\; \hat{\mathbf{v}}(\mathbf{x}, t) \ .$$

In every time step t (or generally every point t where the vector field is evaluated) we determine $\hat{\mathbf{v}}(t)$ for $\mathbf{x}(t)$. The associated optimization problem is linear in the unknowns $\mathbf{v}_i, i = 1 \ldots, |\mathcal{V}|$.

4 Implementation

In this section we discuss the implementation of each stage of our approach. These are firstly, the specification of deformations, secondly, the GPU based linear framework for finding vector fields that minimize our error measures in every time step, and finally, the numerical integration of the shape over time.

4.1 Defining Deformations

Deformations are defined by certain constraints. In our case these are constraints on the vector field $\mathbf{v}(\mathbf{x}, t)$, i.e., for a subset of vertices, we prescribe the associated time-dependent vectors. This way we implement the standard handle metaphor for shape modeling: the user selects surface regions which are either fixed, deformable, or displaced by a handle. In fixed regions, the vector field is constant zero at each time step. The deformable regions define the free parameters, there the vector field is determined by minimization. In handle regions, vertices are displaced over time along smooth curves. The tangents of these curves define the vector field at each handle and hence determine the movement of handle vertices.

From the user's point of view, fixed and handle regions are selected. There is no restriction on the number of such regions or their connectivity. Then she prescribes arbitrary parametric guidance curves for moving the handles. Note that a single curve prescribes the rigid motion of a full handle region: every handle vertex is associated with the curves' tangent.

Such guidance curves can be designed easily and intuitively [15]. Indeed, our approach is even more general in a sense that we are not restricted to parametric curves for specifying constraints: any piecewise linear time-dependent vector field can be applied to prescribe motion of handles. In particular, twisting and bending of the shape can be modeled easily.

4.2 Minimizing Error Measures

Error measures are quadratic forms in the unknown vector field $\mathbf{v} \in \mathbb{R}^{3\,|\mathcal{V}|}$. Our goal is to minimize an expression

$$d(\mathbf{x}, \mathbf{v}) = \mathbf{v}^\top \left[(1-\omega)\,\mathbf{D}_1 + \omega\,\mathbf{D}_2\right]\,\mathbf{v} \ .$$

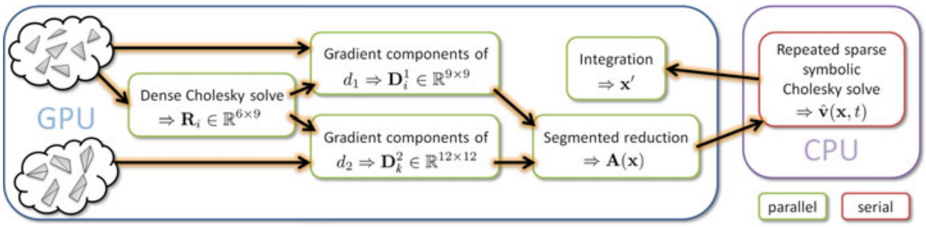

GPU
Dense Cholesky solve
$\Rightarrow \mathbf{R}_i \in \mathbb{R}^{6\times 9}$
Gradient components of
$d_1 \Rightarrow \mathbf{D}_i^1 \in \mathbb{R}^{9\times 9}$
Gradient components of
$d_2 \Rightarrow \mathbf{D}_k^2 \in \mathbb{R}^{12\times 12}$
Integration
$\Rightarrow \mathbf{x}'$
Segmented reduction
$\Rightarrow \mathbf{A}(\mathbf{x})$
Repeated sparse symbolic Cholesky solve
$\Rightarrow \hat{\mathbf{v}}(\mathbf{x}, t)$
CPU
parallel
serial

Fig. 4. Computational Pipeline. For each triangle on the GPU we compute the parameter mapping matrices $\mathbf{R}_i$ which are used to determine the gradient components of the distortion and smoothness measures d_1 and d_2. These coefficients are combined in the final sparse system matrix which is solved by the CPU using a precomputed symbolic factorization yielding the optimal vector field.

where $\mathbf{D}_1$ and $\mathbf{D}_2$ implement error measures d_1 and d_2, respectively. The setup of the corresponding linear system $\mathbf{A}(\mathbf{x})$ is attended by considerable computational costs but is inherent parallizable. To guarantee fast execution times we opt for the combined GPU and CPU approach shown in Figure 4.

Best rigid fields. Both measures d_1 and d_2 depend on the evaluation of the best fitting rigid fields $\hat{\mathbf{r}}_i$ (parameterized by $\mathbf{p}_i = \left(\mathbf{r}_{t,i}^\top, \mathbf{r}_{r,i}^\top\right)^\top \in \mathbb{R}^6$) induced by the linear vector fields $\mathbf{v}_i = \left(\mathbf{v}_r^\top, \mathbf{v}_s^\top, \mathbf{v}_t^\top\right)^\top \in \mathbb{R}^9$ applied to single triangles $T_i = (r, s, t) \in \mathcal{T}$. For each triangle i we therefore find the linear maps $\mathbf{R}_i \, \mathbf{v}_i = \mathbf{p}_i$, $\mathbf{R}_i \in \mathbb{R}^{6\times 9}$ relating linear and best rigid fields in the following way:

Switching to matrix notation (1) is expressed as $e_{T_i} = ||\mathbf{L}_i^\top (\mathbf{v} - \mathbf{E}_i \, \mathbf{p}_i)||^2$. Here $\mathbf{E}_i \in \mathbb{R}^{9\times 6}$ evaluates $\hat{\mathbf{r}}$ at the triangle vertices and $\mathbf{L}_i^\top$ is the Cholesky factor of the matrix $\mathbf{N}_i = \mathbf{L}_i \, \mathbf{L}_i^\top \in \mathbb{R}^{9\times 9}$ performing the integration along the triangle and which is defined by $N_{i,lm} = \begin{cases} \frac{\text{area}(T_i)}{6} & l = m \\ \frac{\text{area}(T_i)}{12} & (l-m) \mod 3 = 0 \\ 0 & \text{else} \end{cases}$. Then (2) is solved for $\mathbf{R}_i$ by solving the linear system corresponding to $\nabla_{\mathbf{p}_i} e_{T_i} = \mathbf{0}$ yielding

$$\mathbf{R}_i \;=\; (\mathbf{E}_i^\top \, \mathbf{N}_i \, \mathbf{E}_i)^{-1} \, \mathbf{E}_i^\top \, \mathbf{N}_i \, .$$

We perform these independent computations in a parallized and numerically stable way for each triangle on the GPU by computing one dense Cholesky factorizations of $\mathbf{E}_i^\top \, \mathbf{N}_i \, \mathbf{E}_i$ and nine corresponding back-substitutions.

Distortion and smoothness gradients. For each triangle i its gradient components $\mathbf{D}_1^i \in \mathbb{R}^{9\times 9}$ contributing to $\mathbf{D}_1$ are found by the evaluation of the gradient of $d_{T_i} = ||\mathbf{L}_i^\top (\mathbf{I} - \mathbf{E}_i \, \mathbf{R}_i) \, \mathbf{v}_i)||^2$ giving

$$\mathbf{D}_1^i = 2\,(\mathbf{I} - \mathbf{R}_i^\top \, \mathbf{E}_i^\top)\, \mathbf{N}_i \,(\mathbf{I} - \mathbf{E}_i \, \mathbf{R}_i)\, .$$

Similarly we find $\mathbf{D}_2^k \in \mathbb{R}^{12\times 12}$ contributing to $\mathbf{D}_2$ for each neighboring pair k of triangles T_i and $T_j = (s, r, t')$ by the evaluation of the gradient of

$f_{T_i,T_j} = ||\mathbf{L}_j^\top (\mathbf{I}' - \mathbf{E}_j \mathbf{R}_i \mathbf{P}_k) \mathbf{v}_{ij}||^2$, where $\mathbf{I}' = [\mathbf{I}, \mathbf{0}]$ and with a permutation matrix $\mathbf{P}_k \in \mathbb{R}^{9\times 12}$ selecting the vector $\mathbf{v}_i$ corresponding to T_i out of $\mathbf{v}_{ij} = \left(\mathbf{v}_j^\top \; \mathbf{v}_{t'}^\top\right)^\top \in \mathbb{R}^{12}$ in the correct order. Computed in parallel on the GPU this yields

$$\mathbf{D}_2^k = 2\,(\mathbf{I}'^\top - \mathbf{P}_k^\top \mathbf{R}_i^\top \mathbf{E}_j^\top)\, \mathbf{N}_j\, (\mathbf{I}' - \mathbf{E}_j \mathbf{R}_i \mathbf{P}_k)\,.$$

Linear systems. In the last step the final sparse symmetric linear system $\mathbf{A}$ is constructed as half of the Hessian of d in parallel by a weighted segmented reduction operation [23]. The non-zero entries are computed of all $\mathbf{D}_1^i$ and $\mathbf{D}_2^k$ by weighted sums according to

$$A_{ef} = \frac{1}{2} \frac{\partial^2 d}{\partial v_e \partial v_f} = \frac{1}{2} \left((1-\omega)\, D_{1,ef} + \omega\, D_{2,ef} \right)\,.$$

We minimize $d(\mathbf{x}, \mathbf{v}) = \mathbf{v}^\top \mathbf{A} \mathbf{v}$ on the CPU subject to boundary conditions (see Section 4.1). $\mathbf{A}$ is symmetric positive definite and sparse with about 1.5% non-zero entries. The linear systems are solved by state-of-the-art direct solvers, namely a sparse Cholesky factorization in combination with an approximate minimum degree preordering to reduce fill-in [9]. We exploit the fact that the structure of the linear system stays fixed in consecutive minimization steps, which allows the precomputation of a symbolic factorization that strongly accelerates the optimization. Experiments reveal that the direct CPU solver is two orders of magnitude faster than a GPU based sparse preconditioned conjugate gradient solver.

4.3 Numerical Integration

In order to compute the deformation we require integration of vertex positions, i.e., we have to find the solution to an initial valued ordinary differential equation. Straightforward Euler integration yields visually pleasing results even for moderate time-steps. However, the lack of accuracy of this scheme would spoil our overall approach and render this scheme unacceptable. Instead we rely on higher order schemes.

For numerical integration we prefer a third order multistep predictor-corrector method with adaptive step size control: in contrast to single-step methods such as Runge-Kutta, the Adams-Bashforth-Moulton scheme (see, e.g., [21]) takes advantage of results from previous integrations steps. The initialization is provided by few fourth order Runge-Kutta steps. The main motivation for choosing this method is the relatively fewer number of expensive evaluations of the vector field required compared to single-step methods when assuming similar approximation errors. This choice was justified by experiments.

5 Analysis and Results

In order to analyze our approach, we apply it to the four standard deformation problems defined by Botsch and Sorkine [6]. In particular, we compare our defor-

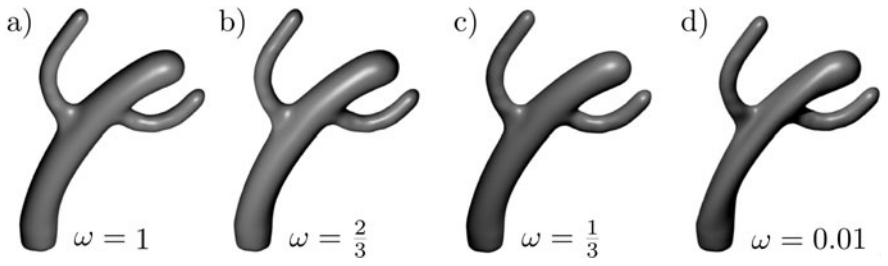


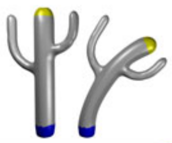
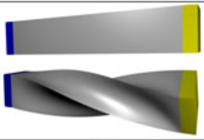
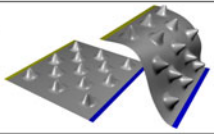
Fig. 5. Weighting of Error Measures. Smaller weights ω lead to small distortion (see table 1) but may produce deformation artifacts if the boundary conditions do not allow for an isometric deformation (d). Slightly higher ω values solve the problem (c).

mations with the (discrete) deformations described in [5] (PriMo), [7] (Thin-Shells), [27] (GradientEd), [24] (LaplacianEd), and [19] (RotationInv). The results are summarized in Table 1. It first shows that our deformations look visually convincing (the images in Table 1 show the original and the deformed shape by using our method with $\omega = \frac{1}{3}$ for each of the four benchmark shapes).

In order to do an additional quantitative comparison, we compute an estimation of the final metric distortion by summing up the squared differences of the edge lengths in the original and deformed mesh (*metricErr*). In a similar way we compute the area distortion by considering the squared area difference over all triangles (*areaErr*), and the angular distortion by considering the squared angle differences over all triangles (*angleErr*). These values are measured between original and deformed shape for the reference deformation, and between original and final time step deformation for our method. For our method, we use four different values of ω. Since the absolute values of the distortion do not have a geometric meaning (because they depend on a particular triangulation), we normalized them by the distortion of our method with $\omega = \frac{1}{3}$. (A number above 100% in the table indicates a higher distortion than for our method with $\omega = \frac{1}{3}$.) For all examples, our technique shows significantly smaller metric distortion than any of the compared discrete deformation techniques. In fact, our continuous approach achieves even lower errors than the discrete, yet non-linear PriMo approach. Moreover, the same statement holds for most of the techniques concerning area and angular distortion. Also note that our approach performs especially well on the bumpplane problem, since only at the small junctions of the bumps to the underlying plane minimal distortions are introduced and the remaining surface deforms isometrically with correct detail orientation.

Timings were measured on a 2.6GHz AMD Opteron system with 8GB RAM and a NVIDIA GTX 280 running CUDA. The number of required integration steps for each problem was 13 (cactus), 44 (bar), 46 (cylinder) and 76 (bump plane), and the number of vector–field evaluations was roughly twice as many. The first three models can be modified at interactive rates. However, performance is impaired for the very large bump plane model. Here the sparse solver becomes the bottleneck of the optimization due to the size of the arising linear system, which was not the case for all other examples in this paper.

Table 1. Experimental Results. The tables summarizes the final errors of the benchmark problems. Measures are given relative to the absolute values of the shape integrated with parameter $\omega = \frac{1}{3}$. The images illustrate the deformation results of our approach using this parameter.

Cactus 70° Bend 5261V 10518T		PriMo	ThinShells	GradientEd	LaplacianEd	RotationInv	New Method: IsometryPreserving $\omega = 1$	$\omega = \frac{2}{3}$	$\omega = \frac{1}{3}$	$\omega = 0.01$	
metricErr	(%)	128	215	199	831	242	101	101	100	90	
areaErr	(%)	125	196	194	739	96	96	95	100	81	
angleErr	(%)	167	272	275	1151	243	101	100	100	93	
Time	(s)						4	4	4	5	
Bar 135° Twist 6084V 12106T											
metricErr	(%)	411	2874	369	2586	333	109	107	100	32	
areaErr	(%)	130	10168	137	6951	878	107	105	100	55	
angleErr	(%)	331	96	298	1230	336	110	107	100	22	
Time	(s)						11	12	13	13	
Cylinder 120° Bend 4802V 9600T											
metricErr	(%)	184	1074	365	783	346	106	104	100	33	
areaErr	(%)	139	715	344	661	33	106	105	100	33	
angleErr	(%)	164	212	340	1016	242	105	104	100	31	
Time	(s)						8	8	9	9	
Bumpplane Translation 40401V 80000T											
metricErr	(%)	21478	24163	694283	366961	508844	118	112	100	10	
areaErr	(%)	18070	20611	513386	280151	458046	116	111	100	12	
angleErr	(%)	12911	11329	383755	654916	531962	116	111	100	13	
Time	(s)						212	241	275	287	

Further examples of our approach are in Figures 2 and 8. Figure 2a shows a perfectly isometry-preserving deformation of a developable surface. Figure 2b (top) shows a twisting deformation of the head of a cow model. There, the body leans forward to compensate metric distortion. Figure 2c shows that the deformations can contain different handle paths: the front legs of the animals were moved in different directions, yielding realistic deformations. In Figure 8 a beetle car model is deformed in four antipodal directions giving four rather different deformation results.

The impact of the variation of the weight ω is illustrated in Figure 5 (error values are listed in Table 1). It shows that too small weights ω in (d) can lead to artifacts when deformations can not be perfectly isometric and metric distortion minimization is enforced at the expense of vector-field smoothness in (6).

6 Discussion

In this section we discuss several aspects of our approach.

Isometry Measure. For our approach, it was necessary to develop a new discrete measure of metric distortion of a vector field acting on the surface. The usual

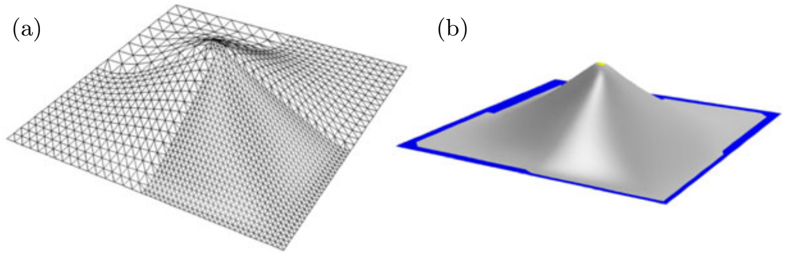


Fig. 6. Mesh Resolution Independence. Pulling the handle vertices up vertically produces a smooth deformation (b) independent of the inhomogeneous mesh resolution (a).

approach (4) as used in [18,11] fails because of two reasons: first, it does not consider the shape and size of the triangles. Figure 3b gives an illustration of this: for this experiment, the surface $z(x,y) = \frac{1}{2}(1+x)(1-x)(1+y)(1-y)$ was sampled over the interval $[-1,1] \times [-1,1]$ as shown in Figure 3b (1–2). Note that an irregular tessellation was chosen. The deformation was defined by keeping the boundary constant and translating the (yellow) region in the direction of the z-axis. Figures 3b (3–5) show three steps of an Euler integration by minimizing $\bar{d}(\mathbf{x},\mathbf{v})$, while Figure 3b (6–8) shows the same steps by minimizing $d(\mathbf{x},\mathbf{v})$ with $\omega = 0$. Even this very small example clearly shows that the measure $\bar{d}(\mathbf{x},\mathbf{v})$ does not yield the desired results. Note that this is not due to missing regularization (the initial surface is not planar), the linear operators are sufficiently well conditioned.

The second reason for developing the new measure is that it offers a simple method to incorporate the smoothness of the surface, i.e., to prevent appearance or disappearance of sharp edges during the deformation. While we do not see a straightforward way to extend (4) in this direction, our measure can easily deal with it as shown in Section 3.3.

We conclude this aspect by visualization tessellation-independence of our deformation method for a simple example. Figure 6 shows a plane that is triangulated with different resolutions and then deformed trivially: the tessellation has no effect on the result due to the design of our measure (see also Figure 3a and Section 3.2).

Handle Path Dependency. The result of our deformation depends not only on the final position of the vertices building the boundary constraints but also on the paths on which they move from starting to final position. This is a significant difference to most existing deformation approaches. Figure 7 illustrates this. There, the shape (a) is deformed by moving the yellow boundary to the right while keeping the blue boundary constant (b). The successive reverse deformation (c) gives almost the original shape. Contrary, moving the handle to the left first (Figure 7 (d)) ends up in a significantly different shape (e). Figures 7 (f–i) show the movement of the handle over two mirrored paths, yielding different final shapes.

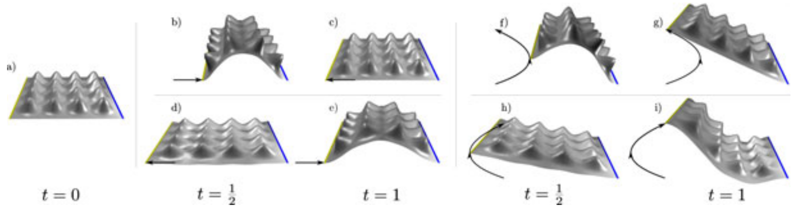


Fig. 7. Handle Path Dependency. Deforming the initial surface at $t = 0$ by different handle pathways from and to the same rest positions result in different deformation results at $t = 1$. The images (b–e) show two linear antipodal deformations, the images (f–i) two parabolic deformation curves.

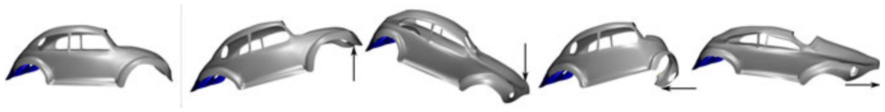


Fig. 8. Beetle Car Deformations. The original beetle model (left) is deformed by fixing the rear of the car and moving the handle at the engine hood into four different directions (right).

While such a path dependence is not always the desired scenario, we believe that for a number of applications it opens a wider flexibility of the modeling process because it reflects the fact that real materials are never totally elastically deformed. The "memory effect" of the deformation gives the look of a combined plastic and elastic deformation of a real material, even though only geometric measures of the surface are considered. Furthermore, it allows to obtain a strong twisting as shown in Figure 2b (bottom) which is impossible by path independent methods.

Comparing to time-dependent Laplacian. We point out that it is not sufficient to modify existent discrete and direct methods to operate in a continuous setting to minimize metric distortion. Consider Figure 9 as an example. For this deformation a bi–Laplacian operator (see [6]) was discretized for every time step, partial deformations were integrated within the same solver, boundary constraints are same as in Figure 6. Comparing results, metric distortion is still significantly higher – it didn't improve much – than for our method. This is not surprising as different errors are minimized. We remark as bottom line that breaking a discrete bi–Laplacian deformation trivially into a "continuous" deformation one cannot achieve the same effect as our continuous method and other discrete state-of-the-art methods are likely to exhibit higher distortion by this strategy compared to our approach, too.

Limitations. We see the main limitation of the approach in the relatively high computation times: despite the fact that we accelerate our approach using the GPU, the technique is far less interactive than state-of-the-art linear frameworks

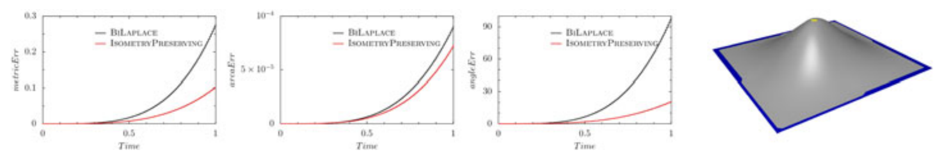


Fig. 9. Time-Dependent bi–Laplacian Deformation. Comparison of metric distortions over time for our method and time-dependent bi–Laplacian deformation.

when applied to large models and thus does not scale to very large meshes yet. We also mention that for the linear operators applied memory footprint is probably higher.

The path dependence of our method can as well be seen as a limitation. We claim it is a features and an integral property of continuous deformations. However, we are aware that depending on the application path dependence may be also interpreted as an artifact.

7 Conclusions

In this paper, we made the following contributions: we defined the deformation by prescribing paths along which certain regions move over time. Then for every time step a piecewise linear vector field is constructed by applying an quadratic energy minimization approach. As a measure for metric distortion, we introduced a new approach which considers the size of the triangles and can be extended to incorporate the surface smoothness. The deformation tries to preserve isometry. It shows significantly lower distortion of length, angles, and area for a set of representative shapes compared to existing standard (linear and non-linear) deformations. Moreover, the results look visually pleasing. Our modeling metaphor defines handle paths. Both the final position of the handles and the path influence the deformation.

The most prominent issue in future research is to further improve the performance. In fact, we see reasonable chances to obtain higher frame rates using a multiresolution approach for the error measures and solvers. Another interesting challenge is the boundary value problem of path planning where the optimal path between two poses is determined.

References

1. Alexa, M., Cohen-Or, D., Levin, D.: As-rigid-as-possible shape interpolation. In: Proc. SIGGRAPH, pp. 157–164 (2000)
2. Angelidis, A., Cani, M.P., Wyvill, G., King, S.: Swirling-sweepers: Constant volume modeling. In: Pacific Graphics (2004)
3. Au, O.K.C., Tai, C.L., Liu, L., Fu, H.: Dual laplacian editing for meshes. IEEE TVCG 12(3), 386–395 (2006)
4. Botsch, M., Kobbelt, L.: An intuitive framework for real-time freeform modeling. ACM Trans. Graphics 23(3), 630–634 (2004)

5. Botsch, M., Pauly, M., Gross, M., Kobbelt, L.: Primo: coupled prisms for intuitive surface modeling. In: SGP, pp. 11–20 (2006)
6. Botsch, M., Sorkine, O.: On linear variational surface deformation methods. IEEE Transactions on Visualization and Computer Graphics 14(1), 213–230 (2008)
7. Botsch, M., Sumner, R., Pauly, M., Gross, M.: Deformation transfer for detail-preserving surface editing. In: Proc. of VMV, pp. 357–364 (2006)
8. do Carmo, M.P.: Riemannian Geometry. Birkhäuser, Boston (1992)
9. Davis, T.A.: Direct Methods for Sparse Linear Systems. SIAM, Philadelphia (2006)
10. Du, H., Qin, H.: Direct manipulation and interactive sculpting of PDE surfaces. CGF (Proc. Eurographics) 19(3), 261–270 (2000)
11. Eckstein, I., Pons, J.P., Tong, Y., Kuo, C.C.J., Desbrun, M.: Generalized surface flows for mesh processing. In: SGP, pp. 183–192 (2007)
12. Efimow, N.: Flaechenverbiegungen in Grossen. Akadmie-Verlag (1957) (German)
13. English, E., Bridson, R.: Animating developable surfaces using nonconforming elements. ACM Trans. Graphics 27, 1–5 (2008)
14. von Funck, W., Theisel, H., Seidel, H.P.: Vector field based shape deformations. In: Proc. SIGGRAPH, pp. 1118–1125 (2006)
15. von Funck, W., Theisel, H., Seidel, H.P.: Explicit control of vector field based shape deformations. In: Proc. Pacific Graphics, pp. 291–300 (2007)
16. Huang, J., Shi, X., Liu, X., Zhou, K., Wei, L.Y., Teng, S.H., Bao, H., Guo, B., Shum, H.Y.: Subspace gradient domain mesh deformation. ACM Trans. Graphics 25(3), 1126–1134 (2006)
17. Kilian, M., Flöry, S., Chen, Z., Mitra, N.J., Sheffer, A., Pottmann, H.: Curved folding. ACM Trans. Graphics 27(3), 1–9 (2008)
18. Kilian, M., Mitra, N.J., Pottmann, H.: Geometric modeling in shape space. ACM Trans. Graphics 26(3) (2007); Proc. SIGGRAPH
19. Lipman, Y., Sorkine, O., Levin, D., Cohen-Or, D.: Linear rotation-invariant coordinates for meshes. In: Proc. SIGGRAPH, pp. 479–487. ACM Press, New York (2005)
20. Nealen, A., Mueller, M., Keiser, R., Boxerman, E., Carlson, M.: Physically based deformable models in computer graphics. CGF 25, 809–836 (2006)
21. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes: The Art of Scientific Computing. Cambridge University Press, Cambridge (2007)
22. Qin, H., Mandal, C., Vemuri, B.C.: Dynamic catmull-clark subdivision surfaces. IEEE Transactions on Visualization and Computer Graphics 4(3) (1998)
23. Sengupta, S., Harris, M., Zhang, Y., Owens, J.D.: Scan primitives for gpu computing. In: GH 2007: Proc. of the 22nd ACM SIGGRAPH/EUROGRAPHICS Symposium on Graphics Hardware (2007)
24. Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C., Seidel, H.P.: Laplacian surface editing. In: SGP, pp. 175–184 (2004)
25. Sumner, R.W., Schmid, J., Pauly, M.: Embedded deformation for shape manipulation. ACM Trans. Graphics 26(3), 80:1–80:7 (2007)
26. Xu, W., Wang, J., Yin, K., Zhou, K., van de Panne, M., Chen, F., Guo, B.: Joint-aware manipulation of deformable models. In: Proc. SIGGRAPH (2009)
27. Zayer, R., Rössl, C., Karni, Z., Seidel, H.P.: Harmonic guidance for surface deformation. CGF (Proc. EUROGRAPHICS) 24(3), 601–609 (2005)

On a (W)ENO-Type Multiscale Representation Based on Quincunx Refinement: Application to Image Compression

Basarab Mateï[1], Sylvain Meignen[2], and Anastasia Zakharova[3]

[1] LAGA Laboratory, University Paris XIII, France
[2] LJK Laboratory, Grenoble University, France
[3] Le2i laboratory, Le Creusot, France
matei@math.univ-paris13.fr, sylvain.meignen@imag.fr,
anastasia.a.zakharova@gmail.com

Abstract. In this paper, we study nonlinear multiscale representations on $\mathbb{R}^2$ which are interpolatory and based on non-diagonal dilation matrices, such as the quincunx matrix. A compression procedure is then introduced for that kind of representations while numerical experiments conclude the paper.

Keywords: Nonlinear Multiscale Representation, Image Compression.

1 Introduction

Due to the hierarchical structure of visual information, multiscale representations are widely used in image processing [5,3,6]. Images are bidimensional complex objects made of homogeneous regions separated by piecewise smooth curves, called edges. Linear multiscale representations based on tensor product wavelet bases are not well adapted to code efficiently an image at its edges. This has motivated the development of new bidimensional image dependent techniques. In order to have a better treatment of the image at its edges, A. Harten in [8,9] introduced a general framework to multiscale data representations. The idea is to associate to any function v, a set of sequences $\mathcal{M}v := (v^0, d^0, d^1, d^2, \ldots)$, where the sequence $v^0 := (v_k^0)_{k\in\mathbb{Z}^2}$ is the coarsest approximation of v and the sequences $d^j := (d_k^j)_{k\in\mathbb{Z}^2}$, with $j \geq 0$, are additional detail coefficients which represent the fluctuations of v between two successive levels of resolution. There exist many ways to build such nonlinear representations. In most papers, the multiscale structure is associated to dyadic levels of resolution [8,5]. We use here the quincunx matrix as dilation matrix to define the scales which allows a two times finer scales discretization than when the tensor product approach is used. Several approaches, using the quincunx matrix as dilation matrix, have already been developed for image representation improving the compression rate [3,7].

In the present paper, we consider a general class of multiscale representations based on a non-diagonal dilation matrix which was already studied in a tensor product approach in [13,5]. We detail the construction of interpolatory and non-linear multiscale representations based on quincunx refinement. More precisely, we study multiscale representations based on affine or quadratic *quasi-linear* prediction operators. We then define a compression procedure which consists in applying the EZW algorithm [16] to a properly renormalized version of the proposed multiscale representation. Using this compression procedure, we show the relevance of using *quasi-linear* prediction operators to define multiscale representations.

2 Notations and Multiresolution Analysis Definition

2.1 Notations

The subspace of bounded sequences on $\mathbb{Z}^2$ is denoted by $\ell^\infty(\mathbb{Z}^2)$, $\|u\|_{\ell^\infty(\mathbb{Z}^2)}$ is the supremum of $\{|u_k| : k \in \mathbb{Z}^2\}$, and $\ell^p(\mathbb{Z}^2)$ is the space of sequences u on $\mathbb{Z}^2$ such that $\|u\|_{\ell^p(\mathbb{Z}^2)} < \infty$, where $\|u\|_{\ell^p(\mathbb{Z}^2)} := \left(\sum_{k\in\mathbb{Z}^2} |u_k|^p\right)^{\frac{1}{p}}$ for $1 \le p < \infty$. For any $w = (w_1, w_2) \in \ell^p(\mathbb{Z}^2)^2$, we define

$$\|w\|_{\ell^p(\mathbb{Z}^2)^2} := \max(\|w_1\|_{\ell^p(\mathbb{Z}^2)}, \|w_2\|_{\ell^p(\mathbb{Z}^2)})$$

We also denote by $L^p(\mathbb{R}^2)$, $1 \le p < \infty$, the space of all measurable functions f such that $\|f\|_{L^p(\mathbb{R}^2)} := \left(\int_{\mathbb{R}^2} |f(x)|^p dx\right)^{\frac{1}{p}}$ for $< \infty$ and $\|f\|_{L^\infty(\mathbb{R}^2)}$ is the essential supremum of $|f|$ on $\mathbb{R}^2$. An invertible matrix M is called a dilation matrix if it has integer entries and if $\lim_{n\to\infty} M^{-n} = 0$ and we put $m := |\det(M)|$. Finally, (e_1, e_2) stands for the canonical basis on $\mathbb{Z}^2$ and for two positive quantities A and B depending on a set of parameters,the relation $A \lesssim B$ implies the existence of a positive constant C, independent of the parameters, such that $A \le CB$. Also $A \sim B$ means $A \lesssim B$ and $B \lesssim A$.

2.2 Multiresolution Analysis of $L^2(\mathbb{R}^2)$

To begin with, let us recall the multiresolution analysis structure associated to a non-diagonal dilation matrix M.

Definition 1. *A multiresolution analysis of $L^2(\mathbb{R}^2)$ is a sequence $(V_j)_{j\in\mathbb{Z}}$ of closed subspaces embedded in $L^2(\mathbb{R}^2)$ satisfying the following properties:*

1. $V_j \subset V_{j+1}$
2. $f \in V_j \Leftrightarrow f(M.) \in V_{j+1}$
3. $\overline{\cup_{j\in\mathbb{Z}} V_j} = L^2(\mathbb{R}^2), \quad \cap_{j\in\mathbb{Z}} V_j = \{0\}$
4. *We assume the existence of a compactly supported function $\varphi \in V_0$, called scaling function, such that the family of their translates $\{\varphi(\cdot - k)\}_{k\in\mathbb{Z}^d}$ forms a Riesz basis for V_0.*

It follows from Definition 1, that the function φ satisfies a dilation equation of type

$$\varphi(x) = \sum_{n \in \mathbb{Z}^2} g_n \varphi(Mx - n), \text{ with } \sum_n g_n = m. \tag{1}$$

We adopt the biorthogonal point of view, that is we assume the existence of a *dual* function $\tilde{\varphi}$ with compact support satisfying

$$\tilde{\varphi}(x) = \sum_{k \in \mathbb{Z}^2} h_k \tilde{\varphi}(Mx - k), \text{ with } \sum_k h_k = m, \tag{2}$$

and such that the following duality property holds $< \tilde{\varphi}(x-n), \varphi(x-k) >= \delta_{n,k}$, where $\delta_{n,k}$ denotes the Kronecker symbol and $< .,. >$ the Euclidean inner product. The approximation at level j can be obtained by projection of $v \in L^2(\mathbb{R}^2)$ on V_j as follows:

$$v_j = \sum_{k \in \mathbb{Z}^2} v_k^j \varphi_{j,k}(x), \tag{3}$$

where, $\varphi_{j,k}(x) = \varphi(M^j x - k)$ and $v_k^j =< v, \tilde{\varphi}(M^j x - k) >$.

3 Interpolatory Nonlinear Multiscale Representations Based on Non-diagonal Dilation Matrix

In this section, we define nonlinear multiscale representations based on a non-diagonal dilation matrix M. Let $(\Gamma^j)_{j \geq 0}$ be the set of embedded grids $\{M^{-j}k, k \in \mathbb{Z}^2\}$. We consider $v^j = (v_k^j)_{k \in \mathbb{Z}^2}$ the data at level j, associated to the location $M^{-j}k$ on the grid Γ^j. In order to build the nonlinear multiscale representation of v, we assume the existence of a *prediction* operator P_j^{j-1} acting from coarse to fine level. This operator computes the approximation $\hat{v}^j = P_j^{j-1} v^{j-1}$ of v^j and may be *nonlinear.*

Interpolatory multiscale representations also assume that $\tilde{\varphi}$ is the Dirac distribution meaning that the data at level j are of the following type:

$$v_k^j = v(M^{-j}k), \quad k \in \mathbb{Z}^2,$$

which implies that $v_k^{j-1} = v(M^{-j+1}k) = v_{Mk}^j$. Thus, to define a nonlinear approximation we only compute $\hat{v}_{Mk+\varepsilon}^j, \varepsilon \in \mathbb{Z}^2 \backslash M\mathbb{Z}^2$ from v^{j-1} using the prediction operator and then compute the prediction error as follows: $d_{Mk+\varepsilon}^{j-1} = v_{Mk+\varepsilon}^j - (P_j^{j-1} v^{j-1})_{Mk+\varepsilon}$, $\varepsilon \in \mathbb{Z}^2 \setminus M\mathbb{Z}^2$. Thus, the data v^j is equivalent to the information contained in (v^{j-1}, d^{j-1}), where $d^{j-1} = \left\{ d_{Mk+\varepsilon}^{j-1}, \varepsilon \in \mathbb{Z}^2 \setminus M\mathbb{Z}^2, k \in \mathbb{Z}^2 \right\}$. Iterating, we obtain the following *nonlinear multiscale representation*:

$$\mathcal{M}v = (v^0, d^0, \cdots, d^j, \cdots). \tag{4}$$

4 Theoretical Results on Nonlinear Multiscale Representations

4.1 Definition of the Prediction Operator

As shown above, the nonlinear multiscale representation is given by the definition of P_j^{j-1}. In the present paper, we will consider an extension of *quasi-linear* prediction operators, defined in [5], to the case where the dilation matrix is non-diagonal:

Definition 1. *A quasi-linear prediction operator S is a function which associates to any $w \in \ell^\infty(\mathbb{Z}^2)$ a linear subdivision operator $S(w)$ defined by*

$$(S(w)u)_k := \sum_{\{l \in \mathbb{Z}^2, \|k-Ml\|_\infty \le K\}} a_{k-Ml}(w)u_l,$$

for any $u \in \ell^\infty(\mathbb{Z}^2)$ and $|a_{k-Ml}(w)| < C$, for any $w \in \ell^\infty(\mathbb{Z}^2)$. The constants K and C are independent of the data w.

The general form for the *prediction* operator, given the data v^{j-1}, is then

$$\hat{v}_k^j = (P_j^{j-1}v^{j-1})_k = (S(v^{j-1})v^{j-1})_k = \sum_{l \in \mathbb{Z}^2} a_{k-Ml}(v^{j-1})v_l^{j-1}. \tag{5}$$

4.2 Definitions of Difference Operators and of Joint Spectral Radius

We say that the *quasi-linear* prediction operator S reproduces the constants when

$$\sum_{p \in \mathbb{Z}^2} a_{k-Mp}(w) = 1, \ \forall k \in \mathbb{Z}^d \text{ and } \forall w \in \ell^\infty(\mathbb{Z}^2). \tag{6}$$

Then, the following result holds:

Proposition 1. *Let S be a quasi-linear prediction operator reproducing the constants. Then, there exists a local and bounded difference operator S_1 such that $\nabla S(w)u := S_1(w)\nabla u$, where $\nabla u_k := (\nabla_i u_k)_{i=1,2} = (u_{k+e_1} - u_k, u_{k+e_2} - u_k)$.*

Proof. Consider

$$\begin{aligned}(S(w)u)_{k+e_i} - (S(w)u)_k &= \sum_{p \in V(k+e_i) \bigcup V(k)} (a_{k+e_i-Mp}(w) - a_{k-Mp}(w))u_p \\ &= \sum_{p \in V(k+e_i) \bigcup V(k)} \alpha_{k-Mp}(w)u_p,\end{aligned}$$

where $V(k) = \{p \in \mathbb{Z}^2, \|k-Mp\|_\infty < K\}$. Since S reproduces the constants, we have $\sum_{p \in V(k+e_i) \bigcup V(k)} \alpha_{k-Mp}(w) = 0$. We then deduce that, since

$$\left\{\nabla_l \delta_{p-\beta}, p \in V(k+e_i) \bigcup V(k), \beta \in \mathbb{Z}^2, l = 1, \cdots, d\right\},$$

spans the sequence orthogonal to the constants and defined for $p \in V(k+e_i)\bigcup V(k)$ [10]:

$$(S(w)u)_{k+e_i} - (S(w)u)_k = \sum_{\beta\in\mathbb{Z}^2} \sum_{p\in V(k+e_i)\bigcup V(k)} \sum_{l=1}^{2} c_{p-\beta,l} \nabla_l u_p^{j-1},$$

where $(c_{.,l})$ is a finite sequence for each l. Computing the differences for other directions e_i we obtain the expected result. □

We will also need the notion of *joint spectral radius* for S_1 in $l^2(\mathbb{Z}^2)$.

Definition 2. *Let S_1 be the difference operator associated to the quasi-linear prediction operator S. Its joint spectral radius in $l^2(\mathbb{Z}^2)$ is defined by*

$$\begin{aligned}\rho_2(S_1) &:= \inf_{j\geq 0} \sup_{(w^0,\cdots,w^{j-1})\in(\ell^\infty(\mathbb{Z}^2))^j} \|S_1(w^{j-1})\cdots S_1(w^0)\|^{\frac{1}{j}}_{(l^2(\mathbb{Z}^2))^2\to(l^2(\mathbb{Z}^2))^2} \\ &= \inf_{j\geq 0}\{\rho, \|S_1(w^{j-1})\cdots S_1(w^0)\nabla u\|_{(l^2(\mathbb{Z}^2))^2} \lesssim \rho^j \|\nabla u\|_{(l^2(\mathbb{Z}^2))^2}, \forall u \in \ell^\infty(\mathbb{Z}^2)\}\end{aligned}$$

4.3 Multiscale Representation Convergence Theorem

We now give a convergence result for the multiscale representation when the prediction operator is *quasi-linear* and when the matrix M is isotropic, corresponding to the following definition:

Definition 3. *The matrix M is isotropic if it is similar to the diagonal matrix $diag(\sigma_1,\sigma_2)$, i.e. there exists an invertible matrix Λ such that*

$$M = \Lambda^{-1} diag(\sigma_1,\sigma_2)\Lambda,$$

$(\sigma_i)_{i=1,2}$ being the eigenvalues of the matrix M and $|\sigma_1| = |\sigma_2| = \sqrt{m}$.

Moreover, for any given norm on $\mathbb{R}^2$ there exist constants C_1, C_2 such that for any integer n and for any $v \in \mathbb{R}^2$

$$C_1 m^{n/2}\|v\| \leq \|M^n v\| \leq C_2 m^{n/2}\|v\|. \tag{7}$$

The convergence theorem we now recall involves $L^2(\mathbb{R}^2)$. For any $v \in L^2(\mathbb{R}^2)$, we can show the following result:

Theorem 1. *If the quasi-linear prediction operator S reproduces the constants, if $\rho_2(S_1) < m^{1/2}$ and if*

$$\|v^0\|_{l^2(\mathbb{Z}^2)} + \sum_{j\geq 0} m^{-j/2}\|d^j\|_{l^2(\mathbb{Z}^2)} < \infty,$$

then the limit function v belongs to $L^2(\mathbb{R}^2)$ and

$$\|v\|_{L^2(\mathbb{R}^2)} \leq \|v^0\|_{l^2(\mathbb{Z}^2)} + \sum_{j\geq 0} m^{-j/2}\|d^j\|_{l^2(\mathbb{Z}^2)}.$$

The proof of this theorem is available in [14] for L^p spaces.

Remark 1. The stability of nonlinear multiscale representations is another crucial issue which is strongly related to that of the associated prediction operator. Since our focus in the present paper is rather to show the relevance of the approach we propose next for image compression from a numerical point of view, the stability issue will not be addressed here and is left for future work.

5 Bidimensional Interpolatory Quasi-Linear Prediction Operators

The nonlinear representations we now introduce are based on (weighted) essentially non-oscillatory ((W)ENO) approaches in an interpolatory framework. (W)ENO methods, first introduced by A. Harten [8], have been widely used to represent piecewise smooth images [1,6]. Such methods are also very useful to study equations coming from the physics such as advection or gaz dynamics [15]. When applied to image processing, these methods follow the general multiscale framework detailed in Section 2 and define the prediction operator P_j^{j-1} depending on some quantity computed on the data v^{j-1} as explained in Section 4.

5.1 Nonlinear (W)ENO Affine Multiscale Representation Based on Quincunx Refinement

In this section, we build an interpolatory multiscale representation based on *quasi-linear* prediction operators and using the quincunx matrix $M = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, to define the scales. One can see this refinement as the insertion of one midpoint per square and two steps are equivalent to quadrisection. Since the scheme is interpolatory, it follows that $v^j_{Mk} = v^{j-1}_k$, and we only have to predict v^j from v^{j-1} at the inserted point in the middle of each square, corresponding to the indices $Mk + e_1$.

We first consider affine interpolation polynomials (i.e. $p(x) = a + bx + cy$) on one of the following four stencils defined on Γ^{j-1}:

$$\begin{aligned} V_k^{j,1} &= M^{-j+1}\{k, k+e_1, k+e_2\} \\ V_k^{j,2} &= M^{-j+1}\{k, k+e_1, k+e_1+e_2\} \\ V_k^{j,3} &= M^{-j+1}\{k+e_1, k+e_2, k+e_1+e_2\} \\ V_k^{j,4} &= M^{-j+1}\{k, k+e_2, k+e_1+e_2\}, \end{aligned}$$

and then define the prediction rules by putting $\hat{v}^j_{Mk+e_1} = p(M^{-j}(Mk+e_1))$, where p is one of the four just defined polynomials. This leads to two potential choices for $\hat{v}^j$, which we denote by $\hat{v}^{j,i}$, $i = 1, 2$:

$$\hat{v}^{j,1}_{Mk+e_1} = \tfrac{1}{2}(v_k^{j-1} + v^{j-1}_{k+e_1+e_2}), \tag{8}$$

$$\hat{v}^{j,2}_{Mk+e_1} = \tfrac{1}{2}(v^{j-1}_{k+e_1} + v^{j-1}_{k+e_2}). \tag{9}$$

To build the *quasi-linear* prediction operator we choose between the two prediction rules (8) and (9) using the following cost function:

$$C^j(k) = \min(|v^{j-1}_{k+e_1} - v^{j-1}_{k+e_2}|, |v^{j-1}_k - v^{j-1}_{k+e_1+e_2}|). \tag{10}$$

When the minimum of $C^j(k)$ corresponds to the first (resp. second) argument the prediction (9) (resp. (8)) is used. We now explain the motivation to use such a cost function. When an edge intersects the cell

$$Q^{j-1}_k := M^{-j+1}\{k, k+e_1, k+e_2, k+e_1+e_2\}, \tag{11}$$

several cases may happen:

1. either an edge intersects $[M^{-j+1}k, M^{-j+1}(k+e_1+e_2)]$ and $[M^{-j+1}(k+e_1), M^{-j+1}(k+e_2)]$ and no direction is favored.
2. or an edge intersects $[M^{-j+1}k, M^{-j+1}(k+e_1+e_2)]$ or $[M^{-j+1}(k+e_1), M^{-j+1}(k+e_2)]$, in this case the prediction operator favors the direction corresponding to the segment not intersected by the edge.

When Q^{j-1}_k is not intersected by an edge, the gain between choosing one direction or the other is very small [7]. We, therefore, need a procedure to define the *edge-cells* Q^{j-1}_k, such that the index k satisfies

$$\begin{aligned} &\operatorname{argmin}(|v^{j-1}_k - v^{j-1}_{k+e_1+e_2}|, |v^{j-1}_{k+e_1+e_2} - v^{j-1}_{k+2e_1+2e_2}|, |v^{j-1}_{k-e_1-e_2} - v^{j-1}_k|)| = 1 \\ &\quad \text{or } \operatorname{argmin}(|v^{j-1}_{k+e_1} - v^{j-1}_{k+e_2}|, |v^{j-1}_{k+2e_1-e_2} - v^{j-1}_{k+e_1}|, |v^{j-1}_{k+e_2} - v^{j-1}_{k-e_1+2e_2}| = 1. \end{aligned} \tag{12}$$

This corresponds to the case where the first order differences are locally maximum is the direction of prediction. When a cell does not satisfy this property, we will apply the rule (9) to predict.

Let us expand the different prediction rules that can be used to predict at level j from levels $j-1$ and $j-2$. Since $M^2 = 2Id$, we have

$$\begin{aligned} &\hat{v}^j_{2k} = v^{j-2}_k, && \hat{v}^j_{2k+e_1} = \begin{cases} \tfrac{1}{2}(v^{j-2}_k + v^{j-2}_{k+e_1}) \\ \text{or } \tfrac{1}{2}(v^{j-1}_{Mk+e_1} + v^{j-1}_{Mk+e_2}) \end{cases} \\ &\hat{v}^j_{2k+e_2} = \begin{cases} \tfrac{1}{2}(v^{j-2}_k + v^{j-2}_{k+e_2}) \\ \text{or } \tfrac{1}{2}(v^{j-1}_{Mk+e_1} + v^{j-1}_{Mk-e_2}) \end{cases}, && \hat{v}^j_{2k+e_1+e_2} = \begin{cases} \tfrac{1}{2}(v^{j-2}_k + v^{j-2}_{k+e_1+e_2}) \\ \text{or } \tfrac{1}{2}(v^{j-2}_{k+e_1} + v^{j-2}_{k+e_2}) \end{cases} \end{aligned}.$$

Looking at the cost function C^j defined in (10), it is clear that the prediction uses the coefficients at level $j-1$ to predict at $2k+e_1$ (resp. $2k+e_2$) at level j only when $|v^{j-2}_k - v^{j-2}_{k+e_1}| = |v^{j-1}_{Mk} - v^{j-1}_{Mk+e_1+e_2}| > |v^{j-1}_{Mk+e_1} - v^{j-1}_{Mk+e_2}|$ (resp. $|v^{j-2}_k - v^{j-2}_{k+e_2}| = |v^{j-1}_{Mk} - v^{j-1}_{Mk+e_1-e_2}| > |v^{j-1}_{Mk+e_1} - v^{j-1}_{Mk-e_2}|$). This suggests that, as we use the smoothest direction to predict, the detail coefficients d^{j-1} shall decay faster than when only the level $j-2$ is used to predict at level j. This will be shown in the Numerical Applications section. Based on this remark, we thus conjecture that the *quasi-linear* prediction operator should lead to a convergent multiscale representation if the simplified multiscale representation associated to the following prediction operator $\tilde{S}$ does:

$$\hat{v}^j_{2k} = v^{j-2}_k, \qquad \hat{v}^j_{2k+e_1} = \tfrac{1}{2}(v^{j-2}_k + v^{j-2}_{k+e_1})$$
$$\hat{v}^j_{2k+e_2} = \frac{1}{2}(v^{j-2}_k + v^{j-2}_{k+e_2}),\ \hat{v}^j_{2k+e_1+e_2} = \tfrac{1}{2}(v^{j-2}_{k+e_1} + v^{j-2}_{k+e_2}). \tag{13}$$

But, as $\tilde{S}$ corresponds to the linear finite interpolation on a type-II triangulation of $\mathbb{R}^2$ with set $\mathbb{Z}^2$, its convergence in $L^2(\mathbb{R}^2)$ is immediate.

The *quasi-linear* prediction operators are associated with interpolation polynomials on different stencils that takes into account the orientation of the edge and lead to so-called essentially non-oscillatory (ENO) multiscale representations. The main drawback of the ENO method is that an arbitrary small change at the round-off level can lead to a change of stencil chosen for prediction. For this reason, the ENO representations are unstable [13] therefore an improvement of these methods, called WENO methods, have been introduced.

The WENO formulation we consider to predict on *edge-cells* (see (12)) is based on a convex combination of potential prediction rules defined above that is: $\hat{v}^j_k := \sum\limits_{r=0}^{m-1} \alpha_{r,k}\hat{v}^{j,r}_k$ with $\alpha_{r,k} \geq 0$ and $\sum\limits_{r=0}^{m-1} \alpha_{r,k} = 1$, which are known to provide a more robust representation than the ENO one. A classical form for the WENO prediction is the following [13]:

$$\hat{v}^j_{Mk+e_1} = \frac{a_{1,k}}{2(a_{1,k}+a_{2,k})}(v^{j-1}_{k+e_1} + v^{j-1}_{k+e_2}) + \frac{a_{2,k}}{2(a_{1,k}+a_{2,k})}(v^{j-1}_k + v^{j-1}_{k+e_1+e_2}) \tag{14}$$

with $a_{1,k} = \frac{1}{\left(\epsilon+(v^{j-1}_{k+e_1}-v^{j-1}_{k+e_2})^2\right)^2}$, $a_{2,k} = \frac{1}{\left(\epsilon+(v^{j-1}_k-v^{j-1}_{k+e_1+e_2})^2\right)^2}$, for some $\epsilon > 0$.

Using the same argument as the one put forward to conjecture that the multiscale representation associated to the ENO prediction operator is convergent, one can build a simplified and convergent version of the WENO multiscale representation as follows:

$$\hat{v}^j_{2k} = v^{j-2}_k,\ \hat{v}^j_{2k+e_1} = \frac{1}{2}(v^{j-2}_k + v^{j-2}_{k+e_1}),\ \hat{v}^j_{2k+e_2} = \frac{1}{2}(v^{j-2}_k + v^{j-2}_{k+e_2}),$$
$$\hat{v}^j_{2k+e_1+e_2} = \frac{a_{1,k}}{2(a_{1,k}+a_{2,k})}(v^{j-2}_{k+e_1} + v^{j-2}_{k+e_2}) + \frac{a_{2,k}}{2(a_{1,k}+a_{2,k})}(v^{j-2}_k + v^{j-2}_{k+e_1+e_2}),$$

and then numerically show that the detail coefficients computed with the prediction operator defined in (14) decrease faster than those obtained with the simplified WENO prediction operator.

5.2 *Quasi-Linear* Prediction Operators Using Higher Degree Polynomials

We now introduce a new type of *quasi-linear* prediction operators based on higher degree interpolation polynomials. In the previous section, we saw that the prediction rules built using an affine polynomial on a given cell are independent of the neighboring cells. Using higher degree polynomials for prediction, on the one hand, would enable to increase the accuracy of the representation on smooth regions but may, one the other hand, create spurious oscillations close to the edges due to the Gibbs phenomenon.

Therefore, the idea developed by some authors is to reduce the degree of the polynomials used to define the prediction operator close to the edges [4]. Our strategy will thus be to use a nonlinear prediction operator based on quadratic polynomials on cells Q_k^{j-1}, defined in (11), which are 4-connected to an *edge-cell* (meaning that either $Q_{k-e_1}^{j-1}$, $Q_{k+e_1}^{j-1}$, $Q_{k-e_2}^{j-1}$ or $Q_{k+e_2}^{j-1}$ is an *edge-cell*), then to use an affine nonlinear prediction operator as defined in section 5.1 on *egde-cells*, and finally to use a linear prediction based on a quadratic polynomial on the remaining cells.

To be more precise, to compute $\hat{v}^j_{Mk+e_1}$ when Q_k^{j-1} is 4-connected to an *edge-cell*, we consider a family of quadratic interpolation polynomials (i.e. $a + bx + cy + dx^2 + ey^2 + fxy$) defined on one of the following four stencils of Γ^{j-1}:

$$\begin{aligned}
V_k^{j,1} &= M^{-j+1}\{k, k+e_1, k+e_2, k+e_1+e_2, k+2e_1, k+2e_2\}\\
V_k^{j,2} &= M^{-j+1}\{k, k+e_1, k+e_2, k+e_1+e_2, k-e_1, k+2e_2\}\\
V_k^{j,3} &= M^{-j+1}\{k, k+e_1, k+e_2, k+e_1+e_2, k+2e_1, k-e_2\}\\
V_k^{j,4} &= M^{-j+1}\{k, k+e_1, k+e_2, k+e_1+e_2, k-e_1, k-e_2\}\,.
\end{aligned} \tag{15}$$

Let us now explain how the stencil selection is carried out. For each stencil $V_k^{j,i}$, we consider the triangles made of neighboring points inside that stencil. This corresponds to 6 triangles for each $V_k^{j,i}$. For instance, for the stencil $V_k^{j,1}$, the triangles are as follows: $M^{-j+1}\{k, k+e_1, k+e_2\}, M^{-j+1}\{k, k+e_1, k+e_1+e_2\}, M^{-j+1}\{k, k+e_2, k+e_1+e_2\}, M^{-j+1}\{k+e_1, k+e_2, k+e_1+e_2\}, M^{-j+1}\{k+e_1, k+2e_1, k+e_1+e_2)\}, M^{-j+1}\{k+e_2, k+e_1+e_2, k+2e_2\}$. We then compute a cost function on each triangle as the sum of the first order differences computed on the triangles that make up $V_k^{j,i}$. For the prediction, we then use the quadratic polynomial associated to the stencil with the minimal cost. On cells Q_k^{j-1} (defined in (11)) that are neither *edge-cells* nor 4-connected to an *edge-cell*, we use the quadratic polynomial defined on $V_k^{j,1}$ to build the prediction operator.

This approach enables the same behaviour as the one using the nonlinear affine prediction operator at *edge-cell* while improves the approximation order as one moves away from the edge. We will see, at the end of this paper, that, doing so, compression results for natural images can be considerably improved. Similarly to the case of *edge-cells* explained in section 5.1, a WENO formulation could easily be derived from the above ENO strategy by making a convex combination of the 4 potential prediction rules on cells 4-connected to an *edge-cell* while of the 2 potential prediction rules on *edge-cell*.

6 Practical Implementation of the Multiscale Representations

We now focus on another aspect of the method which is to create a quadtree structure from the multiscale representation so that it fits into the category of representations that can be compressed with one of the most performant compression algorithm, the so-called EZW (Embedded Zero Tree) algorithm [16].

To obtain a quadtree structure from the multiscale representation, if one considers the level j, the error d^{j-1} is associated to the locations $M^{-j}(k+e_1)$ and $M^{-j}(k+e_2)$ while d^{j-2} is associated to the location $M^{-j}(k+e_1+e_2)$. Looking at Figure 1.(A), it is clear that the error d^{j-1} can again be split into $(\bar{d}^{j-1}, \tilde{d}^{j-1})$ using as the same procedure as the one used to compute (v^{j-2}, d^{j-2}) from v^{j-1}. This leads to the representation of Figure 1.(C).

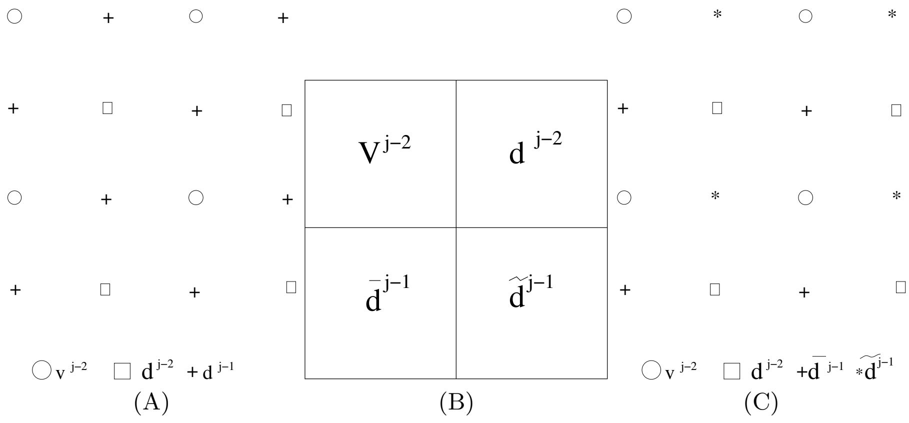


Fig. 1. (A): locations of the points where v^{j-2}, d^{j-2} and $d^j - 1$ are computed,(B): matrix corresponding to two successive steps of decomposition, (C): locations of the points where v^{j-2}, d^{j-2}, $\bar{d}^{j-1}$ and $\tilde{d}^{j-1}$ are computed

The encoding algorithm of a square image of size 2^J thus leads to the representation

$$\mathcal{M}v = (v^0, d^0, \bar{d}^1, \tilde{d}^1, ..., d^{J-2}, \bar{d}^{J-1}, \tilde{d}^{J-1}).$$

This decomposition can be written under a matrix form, which we call V in what follows, as shown on Figure 1.(B), which naturally leads to a quadtree structure.

The EZW algorithm, developed for wavelet transforms, exploits the quadtree structure generated by the wavelet decomposition. The algorithm is based on progressive encoding: the data is compressed through multiple passes with increasing accuracy. The EZW encoder builds zero-tree structures from the quadtree based on the observation that the wavelet coefficients decrease as j increases. Note that this is true provided the wavelet basis is L^2-normalized (i.e. the L^2 norm of the wavelet basis functions should be equal to 1), which is not the case in our interpolatory framework. A very important issue before applying the EZW encoder in our interpolatory framework is thus to renormalize the multiscale decomposition to preserve the decay of the detail coefficients at j increases. In that context, the appropriate renormalization is the following (assuming J is even):

$$\tilde{\mathcal{M}}v = (2^{\frac{J}{2}-1}v^0, 2^{\frac{J}{2}-1}d^0, 2^{\frac{J}{2}-1}\bar{d}^1, 2^{\frac{J}{2}-1}\tilde{d}^1, ..., 2^{p-1}d^{j-2p}, 2^{p-1}\bar{d}^{j-2p+1}, 2^{p-1}\tilde{d}^{j-2p+1}, ..., d^{J-2}, \bar{d}^{J-1}, \tilde{d}^{J-1}). \qquad (16)$$

Considering the matrix V associated to multiscale representation, the initial threshold is then set to $T_0 = 2^{\lfloor \log_2(\max|V_{i,j}|)\rfloor}$. The encoder scans the matrix V using the Morton scan [2], compares it with threshold and gives 'p', 'n', 'z' or 't' as an output; if the absolute value is larger than the threshold, it outputs either 'p', if the coefficient is positive or 'n', if it is negative, else it constructs a tree with the considered element as the root. If it is a zero-tree (that is, the values at the nodes are all smaller or equal to the threshold), the output is 't', and 'z' (isolated zero) otherwise. The EZW encoder assumes that there will be a very high probability that all the coefficients in a quadtree will be smaller than a certain threshold if the root is smaller than this threshold. One then only encodes the elements 'p' or 'n' as outputs. In this case, one puts them in a so-called 'subordinate list' either with $\frac{3T_0}{2}$, if it is larger than $\frac{3T_0}{2}$ or $-\frac{3T_0}{2}$ (for elements smaller than $-\frac{3T_0}{2}$) and remove them from V (replace them by 0) so that they will not be encoded again in that pass. After all the elements have been scanned, the threshold is set to $T_0/2$ and the algorithm starts a new pass and finally stops after a number of passes corresponding to a predefinite minimal value for the threshold. The just described encoding algorithm thus computes a sequence of quantized coefficients $(\hat{v}^0, \hat{d}^0, \hat{\bar{d}}^1, \hat{\tilde{d}}^1, \cdots, \widehat{d^{J-2}}, \widehat{\tilde{d}^{J-1}}, \widehat{\bar{d}^{J-1}})$. We finally write the inverse operator (called decoding step) as follows:

$$\hat{v}^J = \tilde{\mathcal{M}}^{-1}(\hat{v}^0, 2^{-\frac{J}{2}+1}\hat{d}^0, 2^{-\frac{J}{2}+1}\hat{\bar{d}}^1, 2^{-\frac{J}{2}+1}\hat{\tilde{d}}^1, ..., 2^{-2p+1}\hat{d}^{J-2p}, 2^{-2p+1}\widehat{\bar{d}^{J-2p+1}}, 2^{-2p+1}\widehat{\tilde{d}^{j-1}}, ..., \widehat{d^{J-2}}, \widehat{\bar{d}^{J-1}}, \widehat{\tilde{d}^{J-1}}).$$

It is worth noting that due to the compression step, $\hat{v}^j$ is not equal to v^j which implies that the prediction operator at the encoding step and at the decoding step will not be the same. This problem is known as the absence of *synchronization* between the encoder and the decoder. To avoid this problem, a possible choice is to memorize the stencils used at the encoding for the decoding step [7]. However, this requires to allocate bits for that operation which deteriorates the compression results. Our point here is to show that even when the stencils choice is not memorized at the encoding, to use a *quasi-linear* prediction operator in the decoding step improves the compression results.

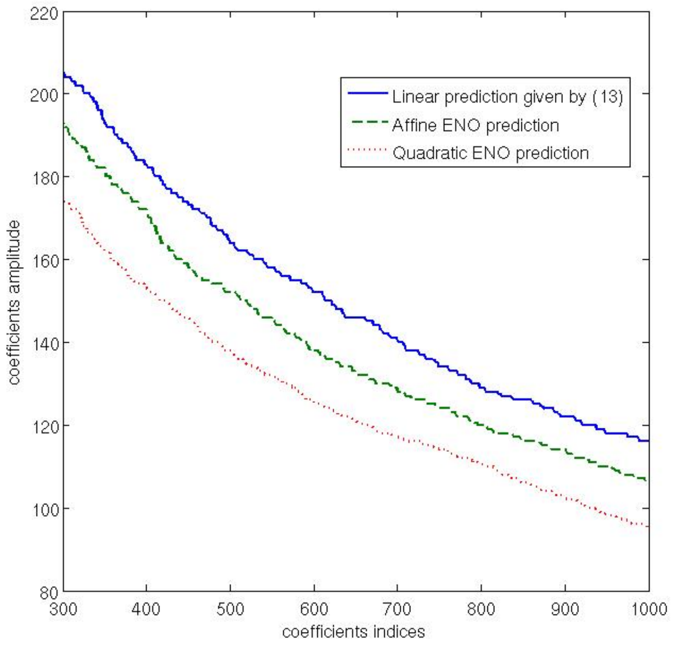


Fig. 2. Decay of the details coefficients for the image of Lena for linear multiscale representation, nonlinear affine representation and nonlinear quadratic representation

7 Numerical Applications

7.1 Decay of the Coefficients

In this section, we compare the decay of the coefficients of the proposed nonlinear decomposition decay compared to that of the coefficients of the linear decomposition, involving only the scale $j-2$ to predict at scale j, given by (13). To do so, we compute the decomposition of the image of Lena until $j = J - 6$ (corresponding to nine subspaces of detail coefficients) either using the ENO predictions based on affine or quadratic polynomials and the prediction based on (13). For each of these 3 cases, we sort the corresponding coefficients in decreasing order. The results displayed on Figure 2 show that the amplitude of the detail coefficients is significantly reduced when one uses a nonlinear prediction operator based on a quadratic or affine polynomial compared to the linear prediction. This suggests that since the corresponding linear multiscale representation is convergent in $L^2(\mathbb{R}^2)$, the proposed nonlinear multiscale representation should also be convergent in that space.

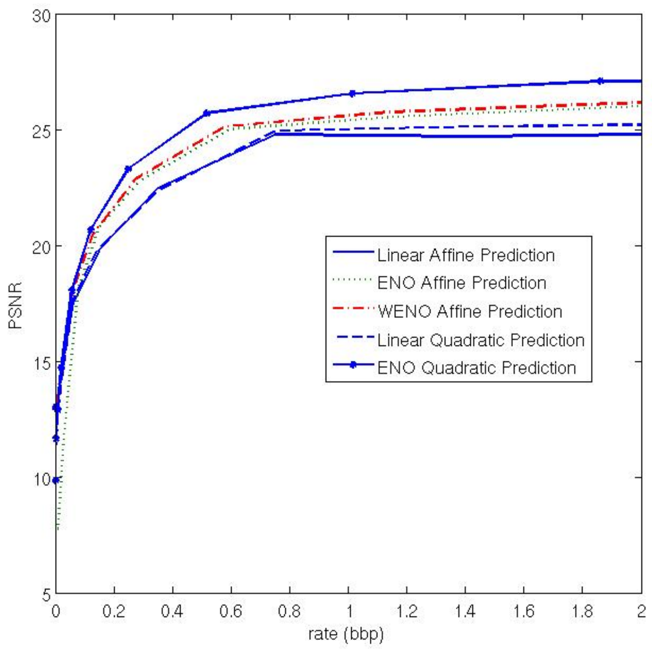


Fig. 3. Compression results for the image of Lena, either using a non separable linear affine multiscale representation, an affine ENO representation, an affine WENO representation ($\epsilon = 10^{-2}$) or an quadratic ENO representation

7.2 Compression Results

To investigate the compression performance of the proposed nonlinear scheme, we compute the nonlinear decomposition until $j = J - 6$ and we then apply the linear scheme for smaller j. The linear scheme corresponds to the application of (9) in the affine case and to consider the quadratic polynomial based on $V_k^{j,1}$ (see section 5.2) in the quadratic case. As on coarse scales edges are not resolved, the nonlinear scheme will be as good as any other scheme therefore we use the linear scheme in that case.

We apply the EZW algorithm to the decomposition of the images of Lena and of Peppers. They both show that the new ENO scheme we propose based on quadratic polynomials representation on smooth regions and affine prediction at edge locations leads to better compression results. These encouraging results suggests that to decrease the degree of the polynomial used to predict at edge location while increasing it when one moves away from the edge is a good strategy. However, proofs of convergence and of stability of such scheme are still missing and will be the motivation for future work.

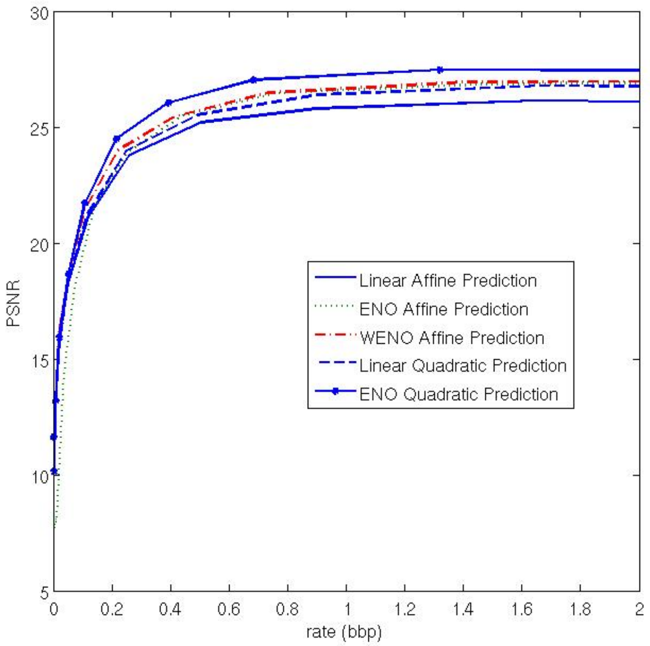


Fig. 4. Similar to Figure 3, but for the image of peppers

8 Conclusion

We have presented some theoretical aspects on nonlinear and non-separable interpolatory multiscale representations. We have then built some particular bidimensional nonlinear multiscale representations based on affine or quadratic interpolatory prediction operators on grids defined using the quincunx matrix. To show the relevance of these approaches, we have proposed an application to image compression. The compression results show a clear improvement brought about by ENO and WENO methods when affine or quadratic *quasi-linear* prediction operator are used compared to linear techniques. The better performance of the prediction based on quadratic polynomials interpolation emphasizes the importance of modifying the order of approximation depending on the image data. In terms of perspectives, we are currently investigating extensions of this approach to non-interpolatory prediction operators and also we are thinking about proofs for both convergent and stability of the proposed multiscale representations.

References

1. Arandiga, F., Cohen, A., Donat, R., Dyn, N., Mateï, B.: Approximation of Piecewise Smooth Images by Edge-Adapted Techniques. ACHA 24, 225–250 (2008)
2. Algazi, V.R., Estes, R.R.: Analysis Based Coding of Image Transform and Subband Coefficients. In: Proceedings of the SPIE, vol. 2564, pp. 11–21 (1995)

3. Amat, S., Busquier, S., Trillo, J.C.: Nonlinear Harten's Multiresolution on the Quincunx Pyramid. J. Comput. Appl. Math. 189, 555–567 (2003)
4. Claypoole, R.L., Davis, G.M., Sweldens, W., Baraniuk, R.G.: Nonlinear Wavelet Transforms for Image Coding via Lifting. IEEE Transactions on Image Processing 12, 1449–1459 (2003)
5. Cohen, A., Dyn, N., Mateï, B.: Quasi-linear Subdivision Schemes with Applications to ENO Interpolation. Appl. Comput. Harmon. Anal. 15, 89–116 (2003)
6. Chan, T.F., Zhou, H.M.: ENO-Wavelet Transforms for Piecewise Smooth Functions. SIAM J. Numer. Anal. 40, 1369–1404 (2002)
7. Chappelier, V., Guillemot, C.: Oriented Wavelet Transform for Image Compression and Denoising. IEEE Trans. Image Process. 15, 2892–2903 (2006)
8. Harten, A.: Discrete Multiresolution Analysis and Generalized Wavelets. J. Appl. Num. Math. 12, 153–193 (1993)
9. Harten, A., Enquist, B., Osher, S., Chakravarthy, S.: Uniformly high Order Accurate Essentially non-Oscillatory Schemes III. J. Comput. Phys. 71, 231–303 (1987)
10. Jia, R.Q., Jiang, Q., Lee, S.: Convergence of Cascade Algorithms in Sobolev Spaces and Integrals of Wavelets. Numer. Math. 91, 453–473 (2002)
11. Amat, S., Donat, R., Liandrat, J., Trillo, J.C.: Analysis of a New Nonlinear Subdivision Scheme Application in Image Processing. Foundations of Computational Mathematics, 193–225 (2006)
12. Amat, S., Liandrat, J.: On the Stability of the PPH Nonlinear Multiresolution. Applied and Computational Harmonic Analysis 18, 198–206 (2005)
13. Matei, B.: Smoothness Characterization and Stability in Nonlinear Multi-scale Framework: Theoretical Results. Asymptotic Analysis 46, 277–309 (2005)
14. Mateï, B., Meignen, S.: A New Formalism for Nonlinear and Non-Separable Multiscale Representations (submitted to Numerical Algorithms, 2011)
15. Serna, S., Qian, J.: Fifth-Order Weighted Power-ENO Schemes for Hamilton-Jacobi Equations. J. Sci. Comput. 29, 57–81 (2006)
16. Shapiro, J.M.: Embedded Image Coding Using Zerotrees of Wavelet Coefficients. IEEE Transactions on Signal Processing 41, 3445–3462 (1993)
17. Amat, S., Busquier, S., Trillo, J.C.: Nonlinear Harten's Multiresolution on the Quincunx Pyramid. Journal of Computational Mathematics 189, 555–567 (2006)

OpenFlipper: An Open Source Geometry Processing and Rendering Framework

Jan Möbius and Leif Kobbelt

Department of Computer Science 8,
RWTH Aachen University,
Ahornstrasse 55, 52074 Aachen
http://www.graphics.rwth-aachen.de

Abstract. In this paper we present OpenFlipper, an extensible open source geometry processing and rendering framework. OpenFlipper is a free software toolkit and software development platform for geometry processing algorithms. It is mainly developed in the context of various academic research projects. Nevertheless some companies are already using it as a toolkit for commercial applications. This article presents the design goals for OpenFlipper, the central usability considerations and the important steps that were taken to achieve them. We give some examples of commercial applications which illustrate the flexibility of OpenFlipper. Besides software developers, end users also benefit from this common framework since all applications built on top of it share the same basic functionality and interaction metaphors.

Keywords: Geometry Processing, Software Framework, Open Source.

1 Introduction

Currently a lot of work is being done to simplify the research and development in the field of geometry processing algorithms. Results of this work are for example the software frameworks Meshlab [2] or Graphite [3]. Most of these frameworks have two major limitations.

On the one hand, they focus on one special type of geometric primitives, namely triangle or polygonal meshes. However there are a lot of applications where multiple types of data are needed, e.g., isosurfaces get extracted from volumetric data or meshes are generated from point clouds. Therefore these applications only focus on parts of the overall geometry processing pipeline. Moreover types like subdivision or spline curves and surfaces are often not supported by these frameworks and thus reduce their versatility. On the other hand, the applications are usually published under the GPL [4] and are therefore not usable in industrial projects leaving out another large group of potential users and contributors.

Our main goal is to provide a common software development platform for the majority of the geometry processing community. Researchers should be able to use the framework to significantly speed up development and focus on the actual research project as most basic functionality is readily available. Industry should obtain a common platform to build their software upon while end users benefit from a unified look and feel for all algorithms using the framework. This large user community heavily improves development as existing code from other people can be re-used and combined to new algorithms.

In the next section we give an overview of our design goals for the OpenFlipper framework. Section 3 briefly describes its central API. In Section 4 the currently available features are presented, and in Section 5 OpenFlipper's scripting system is explained. Finally, some examples on how OpenFlipper simplifies the development process of two projects are given in Section 6.

2 Design Goals

The ultimate motivation for the implementation of a general application framework is that in most research groups various projects are being worked on in parallel and that a considerable percentage of the functionality is shared between those projects. Hence the primary goal of a common framework is to avoid implementation and code redundancy by providing a central repository of modules, e.g., for user interfaces, data file handling, rendering and others.

Another observation is that throughout the life cycle of a project (or of an algorithm) different aspects of software engineering become relevant. In the initial basic research phase, the software framework should allow the programmer to focus on the algorithm itself and it should provide a test bed in which different variants of the algorithms can be evaluated.

At a later stage of the development, the implemented functionality needs to be tested extensively. While the major mode of usage for our framework is that of an interactive application, extensive testing and repetitive execution of similar procedures is more effective in some kind of batch mode. Hence our system comes with a scripting interface that enables the flexible meta-implementation of control procedures that build on top of the available functionality in the C++ implemented modules.

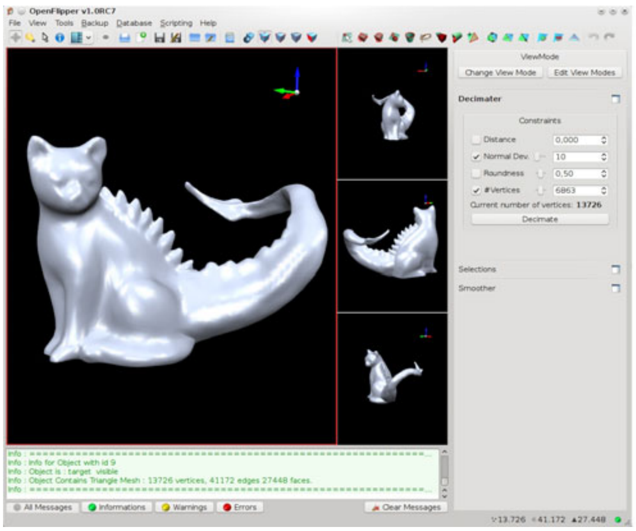
Finally, we want our framework to also support the dissemination and commercialization of geometry processing functionality. This imposes two additional requirements at the technical as well as at the legal level. First, the design of ergonomic and intuitive user interfaces needs to be supported effectively. This is made possible through the option to implement custom designed user interfaces for groups of users with different levels of technical expertise, e.g., software developer or client users. On the legal level we have to resolve licensing issues (this is why we have put our framework under the LGPL license) and we have to offer a mechanism to exchange and license functionality without exchanging source code.

Our solution to the latter issue is that individual modules are implemented as plugins that can be loaded (as pre-compiled code) at runtime. The plugins can access the central functionality through a flexible API that also supports the communication between plugins. In addition we included a license checker mechanism into the framework which enables the protection of individual plugins by identifying the computer on which the program is running.

Our vision is that the OpenFlipper framework will provide a software ecosystem within which an open source community can develop and share their implementations and where there is a seamless transition to and integration with commercial modules. A research group or company could provide some of their plugins as open source while for others a user license has to be obtained. Users can collect their personalized set of plugins satisfying their particular application requirements which can consist of a combination of free and commercial modules.

Currently, the OpenFlipper framework is used intensively in the context of our computer graphics and geometry processing teaching curriculum since it is perfectly suited for student projects.

Compared to other mesh processing software frameworks (e.g., Meshlab [2]) the two major distinctive features are (1) the scripting module which allows users to run OpenFlipper in batch mode and eases the definition of custom tailored interfaces and (2) the scenegraph structure that can handle multiple objects (even with different representations) simultaneously.



Distance 0,000
✓ Normal Dev. 10
Roundness 0,50
✓ #Vertices 6863
Current number of vertices: **13726**
Decimate

```
Info : ==========================================================
Info : Info for Object with id 9
Info : Object is : target  visible
Info : Object Contains Triangle Mesh : 13726 vertices, 41172 edges 27448 faces.
Info : ==========================================================
```

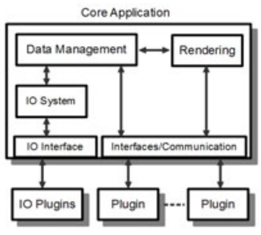


Fig. 1. Left: View of the OpenFlipper interface. It consists of a menu bar and several toolbars at the top, a configurable set of toolboxes on the right, and a large area for the viewers on the left. The viewer area can be split into multiple views rendering the scene with different visualization settings. Every part of the user interface can be extended and re-designed by plugins. Right: Framework scheme.

3 OpenFlipper API

A common design goal for today's software is cross-platform compatibility. As a consequence, a user interface library has to be used which must enable platform independent software development. We chose the *Qt cross-platform application and User Interface framework* [11] and adopted our API to some of the metaphors and concepts used by it.

Qt is based on a signal/slot architecture. Every event, e.g., a mouse click, results in a signal. Every signal can be connected to an arbitrary number of slots which are functions to be executed at each occurrence of the event. Events occuring in parallel are stored in a global event queue and get executed by the main thread in a serialized order.

OpenFlipper uses this signal/slot metaphor as its internal communication paradigm. OpenFlipper's core application communicates with the plugins by emitting signals and in turn activate functions at their counterpart. The functions provided by plugins or by the core application are grouped into separate interfaces which are subdivided into user interaction (keyboard or mouse events, picking), user interface specification (toolboxes, menus), file input/output, object modification, texturing, view control and many others. It is not necessary to implement all functions of an interface, allowing developers to quickly develop little plugins focusing on certain algorithms. The core application will automatically detect what has been implemented and only the relevant signals will be passed to the plugins avoiding function call overhead.

OpenFlipper is not bound to a specific type of geometric data. The core application is independent of the data representation as types are handled via container objects. These objects provide the rendering code and the actual data structure as well as functions for setting and retrieving the data. Due to this representation, arbitrary types can be added by the programmer with little additional implementation effort. For common data types like triangle and polygonal meshes OpenFlipper uses the OpenMesh library [1] which is a highly efficient halfedge based data structure. Additionally, we plan to integrate other commonly used data types like the tetrahedron meshes implemented in CGAL [13] or the volume data acquired with CT or MRI scanners.

The framework does not introduce an abstraction layer between algorithms and data types. Therefore, developers can directly access the underlying data structure (e.g., OpenMesh or CGAL) in their plugins. This results in low effort when porting existing algorithms to OpenFlipper as no code change has to be made for the algorithms themselves. OpenFlipper provides several iterators to retrieve objects in the framework. The iterators can be restricted to fetch objects of a specific data type, e.g., triangle meshes, or to fetch only user selections when working on multiple objects. After retrieval and modification of the data, a plugin has to send a signal and the core application will pass the information to all other plugins and updates the viewers accordingly. It is not the main purpose of the OpenFlipper framework to provide a library with geometry processing algorithms. It only provides the framework to use existing algorithms (such as the excellent CGAL [13] library) and easily test and develop new algorithms in one system. Nevertheless, the free branch contains already a significant number of public domain implementations of state-of-the-art algorithms (see Section 4).

An integral part of the OpenFlipper framework is the rendering. The scene is represented by an OpenGL-based scenegraph and already provides rendering functions for the integrated data types. As almost all graphic cards support vertex buffers we use them for efficient rendering (with cache optimization to increase speed). OpenFlipper uses shaders for advanced rendering techniques but also provides fallbacks, if shaders are not supported by the used hardware. For presentations, the system can render images off-screen with very high resolution. Additionally we will integrate scene export for ray tracers and Sketch 3D (Latex code) in a future release. As the system is completely modular, developers can exchange the rendering for different kinds of data types with their own OpenGL code.

We also integrated a solution for tracing mouse clicks through the scene. The scenegraph implements color picking such that mouse events can be directly mapped to the clicked object or parts of the object, e.g., its faces or vertices.

4 Current Functionality

In this section we present a short overview of the currently available functionality of OpenFlipper and give some preview on functionality that will be added in future releases. The framework runs on Windows, Linux and Mac OS X. We provide IO plugins for OBJ (Wavefront), OFF (Object File Format), STL (stereolithography CAD), PLY (Polygon File Format),BVH (Biovision) and OM (OpenMesh). Furthermore we plan to integrate DAE (Collada), VTK (Visualization Toolkit) and NETGEN (automatic mesh generator, implemented along with tetrahedron meshes) support in one of the next releases.

Currently, OpenFlipper has built-in support for several data types. Triangle and polygonal meshes are supported via the integrated OpenMesh [1] data structure. B-Spline curves and surfaces are represented by proprietary data structures (IO supported through OBJ). Furthermore we provide a skeleton datastructure for skeleton based animations of polygon meshes and some simple primitives like planes, spheres and light sources. For future releases, we plan to integrate tetrahedron meshes (CGAL [13]), point clouds, polygonal lines and volumetric data (e.g., from CT or MRI).

For the current release a set of standard geometry processing algorithms is already included. The standard algorithms are implementations from recent research papers and constitute a basic repository for other software developers and users. The OpenFlipper framework only provides the interface to use them. Some of the algorithms have been developed in well maintained libraries such as CGAL and OpenMesh and are only imported here to show how simple it is to integrate new functionality by using existing code. The following list gives a short overview of the available functionality:

Selections: OpenFlipper comes with a large set of selection metaphors for all supported data types. The build in selections are: single click , surface and volume lasso, paint ball, boundary, connected component, and a flood filling based on normal deviation between adjacent primitives. The selection metaphors are implemented for triangle and polygon meshes and, where applicable, for other data types. For polygonal meshes we also distinguish between standard selections and special feature selections (e.g., important edges). For area based algorithms it is possible to select a handle and a modeling area.

Isotropic Remesher: The isotropic remesher implements the remeshing algorithm by Botsch and Kobbelt [6] and tries to establish an average target edge length specified by the user on an isotropic triangle mesh.

Decimater: The decimater based on [15,16] decimates triangular meshes with respect to different constraints for the resulting meshes. It currently supports approximate distance constraints(error quadrics) to the original surface, upper bounds for the normal deviation, aspect ratio of triangles and a target complexity.

Smoother: This plugin implements a Laplacian smoother for triangular meshes [14]. It can smooth in both tangential and normal direction with C^0 or C^1 continuity while keeping the result within a prescribed distance to the original surface. OpenFlipper's feature selection can be used to specify edges or areas which are kept fixed.

Subdivider: Implementation of Loop [8], sqrt(3) [7], interpolatory sqrt(3) [9] and modified butterfly[10] subdivision for triangular meshes.

Mesh Information: This plugin provides various information about arbitrary polygonal meshes. It computes statistics about aspect ratios, dihedral angles, bounding boxes, average edge length, genus, number of elements and many more.

Laplace, Mean Curvature, Gaussian Curvature: The three plugins compute the laplace vector, mean curvature and Gaussian curvature for every vertex on a triangular mesh. The computed values are attached to the mesh and can be used by other plugins.

Texture Control: This plugin provides a user interface to change texture files and to select an arbitrary property on a mesh to be used as texture coordinates. For example the Gaussian curvature plugin computes the curvature at each vertex and stores it as a property. Texture control reads these precomputed values and uses them as texture coordinates for a 1D texture. It also computes a histogram of the values and provides the user with additional functions like clamping values or taking the absolute value of them.

Slicing: Clipping planes can be created by this plugin, enabling the user to look inside of objects. This is especially useful when analyzing tetrahedron meshes like the ones generated by CGAL [13].

Topology Control: This plugin implements elementary topological operations for meshes like flipping or splitting edges and adding or removing faces.

Scripting: This scripting plugin controls the integrated scripting environment. It collects functions available to scripting and provides a script editor. Section 5.1 gives a more detailed description of OpenFlipper's scripting capabilities.

Visual Scripting: The visual scripting plugin provides a higher level interface to the scripting module. This plugin is further described in Section 5.2.

5 Scripting

OpenFlipper provides a variety of modules to ease the development of an interactive application. At a later stage of software development extensive testing of algorithms is required. OpenFlipper provides a scripting environment integrated into the framework to automate such processes in a batch mode. Qt already ships with an excellent scripting system that is used as the basis for OpenFlipper's scripting environment. As the language follows the ECMA-262 standard [12] which is also used for JavaScript, the syntax is familiar to a large number of developers.

The scripting system of OpenFlipper can be divided into two major parts. The first one is the general *text based scripting* system for experienced developers while the second one is a high level *Visual Scripting Interface* built on top of the general scripting.

5.1 General Scripting

OpenFlipper includes a text based scripting editor and interpreter. The editor collects and shows all available functions exported by the plugins, a description of their functionality and parameters. Scripts are visualized with syntax highlighting and can be directly executed without any compilation. Each function provided by a plugin can be made available to the scripting system.

Basic types like vectors or matrices are known to the system and can be used or manipulated directly. Like JavaScript, the language provides loops, conditionals, input/output and many other standard operators. As scripts are evaluated at runtime, all existing algorithms in OpenFlipper can be used and controlled via the scripting system without having to recompile code. This is especially useful when testing and evaluating algorithms or trying to find optimal parameters for a set of algorithms.

The scripting system is also capable of modifying and extending the user interface. Qt includes the Qt Designer tool which can be used to generate user interfaces and toolboxes. The user interface specification files generated by this graphical designer tool can be loaded at run time and connected to all existing algorithms via the scripting language. Consequently no change to a plugin is necessary for creating a new interface, a simple script is sufficient. The quadrilateral remeshing application described in Section 6.1 uses this option.

5.2 Visual Scripting Interface

Scripting is a powerful tool to combine simple algorithmic blocks to more complex algorithms. However, a programming or scripting language is usually too complex for end users. To support less technically-experienced users in generating scripts, we implemented the more abstract Visual Scripting Interface [21]. The Visual Scripting Interface is build directly on top of OpenFlipper's text based scripting system. The visual script editor is a data flow based block editor inspired by the block or filter based processing in audio or video processing applications [20]. The algorithms in OpenFlipper are represented as blocks with separate inputs and outputs. A simple example is an isotropic remesher. Input and output for this algorithm are surface triangle meshes. Additionally there can be other input parameters like, e.g., average edge length for the remeshed output. Figure 2 shows a simple example for such a visual script. This script consists of only three blocks. The first block computes the average edge length of an input mesh. Afterwards the computed length is passed to a math block and divided by a user specified number. The result is then passed to the isotropic remesher that uses the input value as the target edge length for its output mesh. The execution order for the different blocks can get fairly complicated, so the user has to define an order in which the algorithms are called. This is visualized by the data flow connections. For blocks that don't change objects (e.g., math blocks) the execution order is computed automatically.

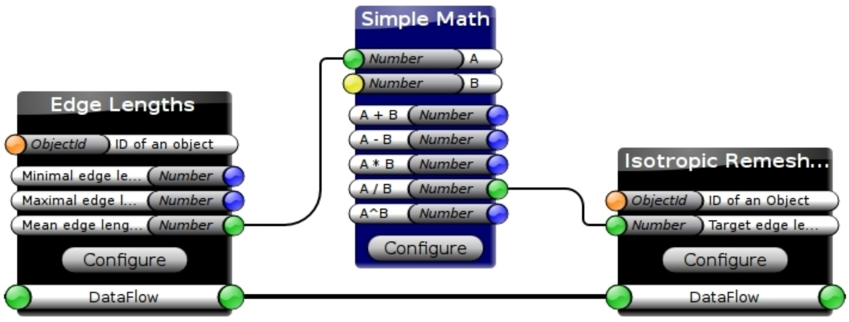


Fig. 2. Remeshing algorithm in the Visual Scripting Editor

Many algorithms require user interaction to select an object to work on. The visual scripting system provides several interactive blocks, allowing to select objects or asking for user input.

From the implementational point of view every block in the editor is associated with an xml file containing in- and output specifications as well as small code snippets which represent the blocks in the final script. Every visual script is therefore parsed, the blocks from the xml files and all variables are connected to a documented OpenFlipper script which optionally can be viewed and modified by the user in the text based script editor. We have observed that these generated scripts provide an excellent foundation to learn the OpenFlipper scripting language. Users can read the code and use it as a starting point for creating more complex algorithms. The scripts are documented and changes made can be directly executed to get new results.

As the scripting blocks are defined and composed from simple xml files, the visual scripting system is not restricted to OpenFlipper's language. Therefore the editor and its components can be used to create script code for arbitrary languages.

6 Industrial Use Cases

A major design goal for OpenFlipper was to provide a toolkit allowing us to reduce the time and implementational overhead when converting research code to an end user application. This requires a solid base of working code that can be legally used in commercial and open source projects (LGPL). Researchers are provided with a stable toolkit, enabling them to focus on implementation and testing of new algorithms while visualization, selection or analytic tools are readily available.

Additionally, the user interface in research and end user applications is usually quite different. As OpenFlipper allows us to create an additional interface on top of the existing functionality, only little effort has to be spend to abstract the interface while keeping the original interface available for expert users.

There are already several commercial projects using OpenFlipper as their platform. The projects provide continuous improvements and new features to OpenFlipper's freely available parts and algorithms. A lot of basic algorithms are implemented for these projects which will also be published as open source and are therefore usable and valuable for the community. In the following sections, we present two of these projects where OpenFlipper significantly reduced the coding effort during development.

6.1 Quadrilateral Remeshing

Based on the OpenFlipper toolkit, a software for generating high quality quadrangular meshes from unstructured triangle meshes [17] has been developed in the context of a commercial project. The user can control the algorithm's output by providing additional constraints for the output structures and therefore modify the resulting quadrangulation.

In the development process of this project the interaction metaphors already defined in OpenFlipper have been used to define these constraints. One of the constraint controls is OpenFlipper's selection system. The user can select important edges and the algorithm uses the selection as a guidance for the final edge directions. Additionally, OpenFlipper provides freely drawable polygonal lines on surfaces which are also used by the system to control the final output. Furthermore the selection system is used to specify where singular vertices should be positioned. All interaction metaphors, visualization, data types and input/output functions for this algorithm were already provided by the toolkit.

At its frontend the algorithm makes extensive use of multiple interfaces. The implementation consists of several parts which are implemented in independent plugins. The first plugin computes the principal curvature directions on the input mesh. The second plugin computes, based on the principal directions and possible user hints, a direction guiding field which is used to control the edge flow in the resulting quadrangular mesh. The third plugin generates a parametrization and extracts a quadrangular mesh. The interfaces to all plugins are available to the professional user while a simple unified interface exists showing only the relevant steps and settings while hiding the remaining parameters (with empirically derived defaults) from the user. This additional interface is purely defined as an OpenFlipper script and can be loaded and even modified at run time.

The mixed-integer quadrangulation solver used in [17] is also freely available as a separate library (CoMISo [18]). An example for the algorithm's output is shown in Figure 3.

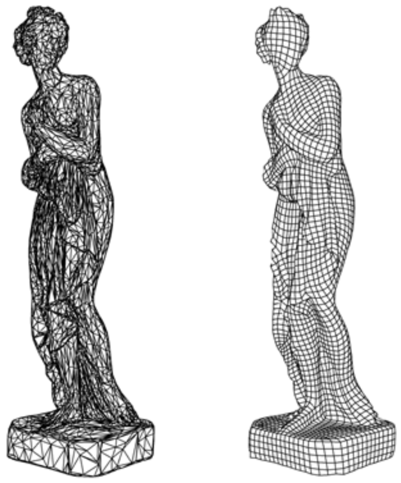


Fig. 3. Model of Iphigenie as an unstructured triangle mesh and the result after the quadrangular remeshing algorithm

6.2 Car Modeling

In this project [19], a semi-automatic approach to efficiently and robustly recover the characteristic feature curves of a given free-form surface has been developed where the input is not required to be a proper manifold. The technique supports a sketch-based interface implemented in OpenFlipper where the user must roughly sketch the feature location by drawing a stroke on the input mesh. For this type of interaction OpenFlipper's picking system provides functions that return the

3D position of a mouse click in the scene. The system then snaps the initial sketch curve to the correct position based on a graph-cut optimization scheme that takes various surface properties into account. Additional positional constraints can be placed and modified manually which allows interactive feature curve editing. The feature curves can be used as handles for surface deformation, since they describe the main characteristics of an object. The system allows the user to manipulate a curve while the underlying surface adopts itself to the deformed feature.

During development of this project a lot of the existing functionality of OpenFlipper has been used and therefore significantly reduced the coding effort for this project. No rendering code was required as it was already available for the B-Spline and mesh data types used in this project. For these types the IO and file management already existed in the framework. Figure 4 shows an example for modeling a car using the final application.

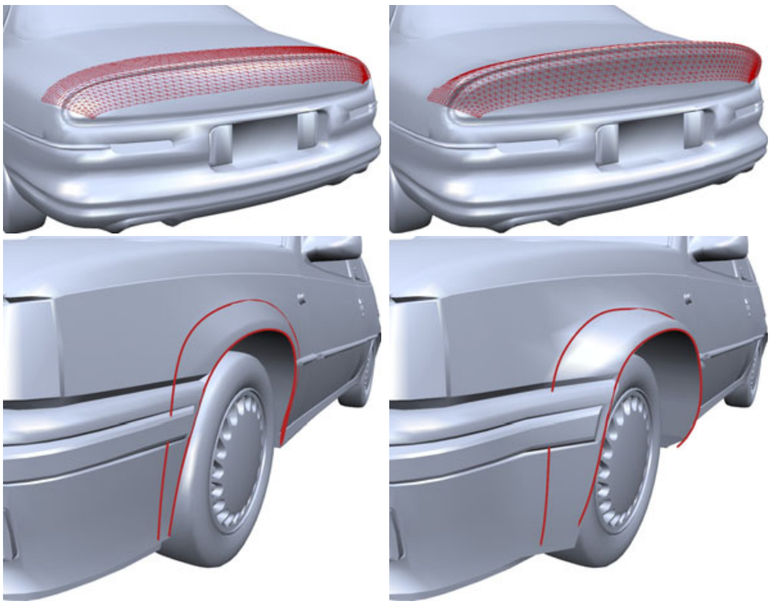


Fig. 4. Car modelling implemented on top of OpenFlipper. Left: Original models, Right: Modified models.

7 Conclusion and Future Improvements

In this paper we presented our OpenFlipper software framework. We developed a system which is should be helpful for the majority of the geometry processing community.

By now OpenFlipper is used for many different purposes. Students use it for learning the basics of computer graphics and especially geometry processing and rendering. Researchers use it to develop and test algorithms while companies benefit by directly using research code in an end user application with only little effort. All of these users provide updates and extensions to the freely available part of the framework. Therefore the functionality of OpenFlipper rapidly increases and it has a growing toolbox of basic algorithms, interaction metaphors and rendering functions which has been given back to the community by the various contributors as free software.

The user community is currently developing a lot of new functions for OpenFlipper. These include support for more file formats (Netgen, VTK, Collada), new data types like point splats or tetrahedron meshes and many algorithms working on them. Figure 5 shows a preview of the tetrahedron meshes.

All this functionality simplifies development and enables the people to focus on their creative work when inventing new algorithms, making OpenFlipper a valuable framework for the community.

The source code and executables are available at our website [22].

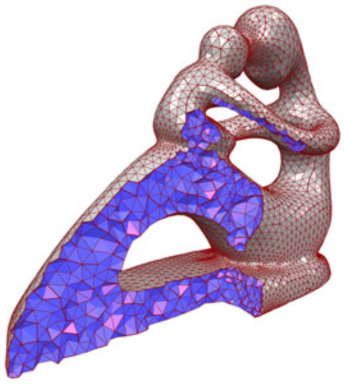


Fig. 5. Preview of a tetrahedron mesh in OpenFlipper

Acknowledgements. We would like to thank all contributors, users, and developers who support the development of OpenFlipper.

References

1. Botsch, M., Steinberg, S., Bischoff, S., Kobbelt, L.: RWTH Aachen: OpenMesh - A generic and efficient polygon mesh data structure. In: OpenSG Symposium (2002)
2. Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G.: Meshlab: an open-source mesh processing tool. In: Sixth Eurographics Italian Chapter Conference, pp. 129–136 (2008)
3. Graphite, `http://alice.loria.fr/index.php/software.html`
4. GNU General Public License, `http://www.gnu.org/licenses/gpl.html`
5. GNU Lesser General Public License, `http://www.gnu.org/licenses/lgpl.html`

6. Botsch, M., Kobbelt, L.: A remeshing approach to multiresolution modeling. In: SGP 2004: Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, pp. 185–192 (2004)
7. Kobbelt, L.: Sqrt(3)-Subdivision. In: Proc. ACM SIGGRAPH 2000, pp. 103–112 (2000)
8. Loop, C.T.: Smooth Subdivision Surfaces Based on Triangles. M.S. Thesis, Department of Mathematics, University of Utah (1987)
9. Labsik, U., Greiner, G.: Interpolatory sqrt(3)-Subdivision. Comput. Graph. Forum (2000)
10. Zorin, D., Schröder, P., Sweldens, W.: Interpolating Subdivision for Meshes with Arbitrary Topology. In: Proceedings of Siggraph, pp. 189–192 (1996)
11. Qt cross-platform application and UI framework, http://qt.nokia.com
12. Standard ECMA-262, ECMA Script Language Specification, 5th edn. (December 2009)
13. CGAL: Computational Geometry Algorithms Library, http://www.cgal.org
14. Botsch, M., Pauly, M., Kobbelt, L., Alliez, P., Lévy, B., Bischoff, S., Rössl, C.: Geometric modeling based on polygonal meshes. In: SIGGRAPH 2007: ACM SIGGRAPH 2007 courses (2007)
15. Kobbelt, L., Campagna, S., Seidel, H.: A General Framework for Mesh Decimation. In: Proceedings of Graphics Interface, pp. 43–50 (1998)
16. Garland, M., Heckbert, P.: Surface simplification using quadric error metrics. In: SIGGRAPH 1997: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques 1997, pp. 209–216 (1997)
17. Bommes, D., Zimmer, H., Kobbelt, L.: Mixed-integer quadrangulation. In: ACM SIGGRAPH 2009 papers, pp. 77:1–77:10 (2009)
18. CoMISo: Constrained Mixed-Integer Solver library, http://www.graphics.rwth-aachen.de/comiso
19. Dekkers, E., Kobbelt, L., Pawlicki, R., Smith, R.: A sketching interface for feature curve recovery of free-form surfaces. Computer-Aided Design (in Press, 2011)
20. Pavic, D., Schönefeld, V., Krecklau, L., Habbecke, M., Kobbelt, L.: 2D Video Editing for 3D Effects. In: VMV 2008, pp. 389–398 (2008)
21. Kasprzyk, D.: Diploma Thesis on Optimized User Interface for Geometry-Algorithms. RWTH Aachen University (2009)
22. OpenFlipper website, http://www.openflipper.org

Parameterization of Contractible Domains Using Sequences of Harmonic Maps

Thien Nguyen and Bert Jüttler

Johannes Kepler University, Linz, Austria

Abstract. In this paper, we propose a new method for parameterizing a contractible domain (called the computational domain) which is defined by its boundary. Using a sequence of harmonic maps, we first build a mapping from the computational domain to the parameter domain, i.e., the unit square or unit cube. Then we parameterize the original domain by spline approximation of the inverse mapping. Numerical simulations of our method were performed with several shapes in 2D and 3D to demonstrate that our method is suitable for various shapes. The method is particular useful for isogeometric analysis because it provides an extension from a boundary representation of a model to a volume representation.

1 Introduction

Parameterization is one of the classical topics and attracts a lot of research in computer graphics and geometric modeling because it plays a central role in various applications such as texture mapping or morphing, to name a few. Even though there are many powerful methods to deal with this problem for surfaces, only few results have been achieved for volumes. In fact, due to computational complexity, volume parameterization is considered a big challenge. In general, methods proposed for this problem exploit information about representation and topology of volumes such as tetrahedral mesh volumes or swept volumes. Our work investigates more general problems in which objects are given in boundary representation.

Assume that we are given a contractible domain defined by its boundary. We also assume that we are able to divide the boundary into several patches, i.e., four curves in plane or six surfaces in three-dimensional space. Our goal is to construct a spline representation for the domain. To achieve that, our method creates a mapping from the domain to the unit square or unit cube, see Figure 1. The mapping should be regular or injective. We attain this by solving several variational problems. Finally, the spline approximation for the inverse of the mapping is found by least squares fitting.

2 Related Work

Isogeometric Analysis. (IGA for short) was invented by Hughes et al. [12] in 2005 as a promising approach to bridge the gap between numerical simulation and computer aided design (CAD). Immediately, it attracted a lot of

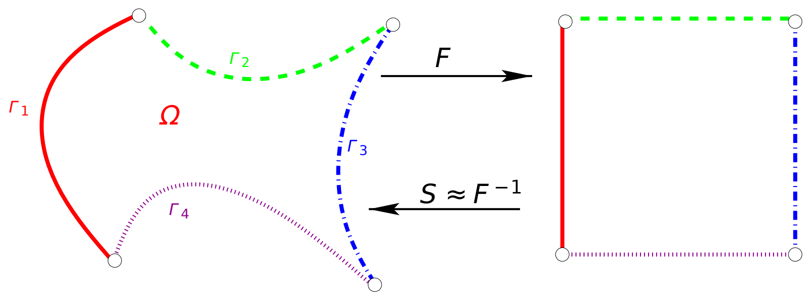


Fig. 1. Mapping from a planar domain to the unit square: the same style curves should be mapped to each other

research interest from both numerical analysis and geometric modeling communities because of some important advantages. The first advantage of IGA is that the computational domains are solid objects which are represented by tensor product B-splines or NURBS. It is superior to classical finite element methods which work on approximate domains represented by discrete meshes. The second advantage of IGA is that the discretization space is spanned by B-splines or NURBS basis and therefore it provides refinement possibilities as proposed in [12]. So, refinement does not require interaction with the original CAD model as finite element methods, but only requires simple operators such as knot insertion or degree elevation. Due to these properties, the authors called the method isogeometric analysis, i.e., geometry is preserved at all levels of refinement.

Nevertheless, there are some difficulties arising when one wants to use IGA to solve a particular problem. One of the problems is that IGA requires solid models represented by tensor product NURBS, i.e., a NURBS mapping from a parametric domain to a physical domain should cover the entire domain. However, CAD systems usually only provide information on the boundary, i.e., mappings from the parametric domains to the boundary of the models [3]. This observation motivates our research to find a spline representation of the solid models which are only defined by their boundary representation.

Ad Hoc Parameterization. The problem of parameterizing a domain defined by boundary curves (or surfaces) can be traced back to the well known work by Gordon and Coons [11]. This method uses blending functions to define a Boolean sum operator interpolating boundary curves. It provides reasonable parameterization for quadrilateral patches with nice shapes. Inspired by this direction, there is a later work by Farin on discrete Coons patches [7]. However, because of its simplicity this method may not be suitable for singular cases and complicated shapes. Recently, Xu et al. have used the discrete Coons method combined with shape optimization method to find optimal parameterizations as a prerequisite of IGA for planar shapes [18].

In volume parameterization motivated by IGA, to the best of our knowledge, there are only two remarkable results. The first is the paper by Martin et al. [15] where the authors proposed a parameterization method for a generalized cylinder-type volume defined by a tetrahedral mesh. This method bases on discrete volumetric harmonic functions and solves several Laplace equations. After remeshing the tetrahedral mesh to the hexahedral mesh, a trivariate B-spline for the solid model is generated by an iterative fitting method. The second is the paper by Aigner et al. [1] where the authors proposed a parameterization method for swept volumes which cover many free-form shapes in CAD system like blades or propellers. In this method, a spline approximation for a solid model can be found by solving a minimization problem with several penalty terms corresponding to particular features of the shape.

In Voruganti et al. [17], a method is proposed to build a bijective map between a genus-zero domain in three-dimensional space and the unit sphere, which partially inspired our approach. The authors first choose a point inside the domain, called "shape center", and compute a potential function by solving the Laplace equation with Dirichlet boundary conditions which are zero at the shape center and one on the boundary. Then, streamlines, which are the curves joining each boundary point to the shape center, are build so that they intersect the level sets of the computed function orthogonally. This results in a parameterization of the unit sphere by means of spherical coordinates.

3 Harmonic Maps

Recently, there has been much research interest in harmonic maps and their applications in computer graphics [2,5,8]. Harmonic maps are used to find mappings between two manifolds due to their beautiful properties. It is well-known that any conformal mapping between 2-dimensional manifolds is harmonic. Moreover, if we consider the mappings between a surface in $\mathbb{R}^3$ and a fixed boundary domain in $\mathbb{R}^2$, harmonic and conformal mappings are the same [10]. A map u between Riemannian manifolds (M, g) and (N, h) is called *p-harmonic* if it is a critical point of the *p-harmonic* energy

$$E_p(u) = \int_M ||\nabla_M u||^p \mathrm{d}volM$$

where $||\nabla_M u||$ is the length of the differential in M. If $p = 2$, we call it simply harmonic map. Results concerning existence and regularity of the solution of the above problem are described by Jost [13]. In particular, we are interested in the maximum principle of the solution in the case that M has a boundary and $N = \mathbb{R}$. Then if u is a harmonic map, it attains maxima or minima values on the boundary. So, we see that the level sets of u on M are contractible hypersurfaces. In our method, we use the following particular cases of manifolds M and N:

1. $M = \bar{M}_0$ and M_0 is an open subset of $\mathbb{R}^n$ ($n = 2$ or 3) and $N \subseteq \mathbb{R}$ (both M and N with Euclidean metrics): In this case, the harmonic energy has the form
$$E_2(u) = \int_M ||\nabla u||^2 \mathrm{d}\mathbf{x}$$

where ∇ is the gradient operator and integration has the usual meaning in $\mathbb{R}^n$.

2. M is a Riemannian manifold in $\mathbb{R}^n$ ($n = 2$ or 3) which is given by the zero level set of some functions, and $N \subseteq \mathbb{R}$: In this case, the harmonic energy has the form
$$E_2(u) = \int_M ||\nabla_M u||^2 \mathrm{d}volM$$
Here, $\nabla_M u$ is the covariant derivative of u with respect to M which is simply the projection of gradient of u in $\mathbb{R}^n$ on the tangent space of M [2].

4 Our Framework

For the sake of simplicity, we describe our framework only for the planar case. It can be naturally extended to the three-dimensional case by similar ideas. Recall that we are given a contractible computational domain Ω defined by a closed piece-wise smooth curve. Assume that we are able to divide the curve into 4 curves. In fact, this can be done simply by picking 4 points on the curve. Our framework consists of 2 main steps.

4.1 Step 1: Finding a Mapping $F : \Omega \to [0, 1]^2$

The mapping F should map the boundary of Ω to the boundary of the unit square and might provide constant parametric speed along boundary. In coordinates, F can be written as $F = (f, g)^T$. In order to use the mapping F in step 2, F should be regular or injective. In other words, the determinant of the Jacobian of F should not vanish in Ω or equivalently, ∇f and ∇g should be linearly independent on Ω. In order to achieve this goal, we propose a 2-step method, namely, we first find f then use f to find g.

Step 1.1. The scalar function $f : \Omega \to [0, 1]$ can be found by minimizing its Dirichlet energy
$$E_2(f) = \int_\Omega ||\nabla f||^2 \mathrm{d}\mathbf{x} \to \min_f \tag{1}$$
subject to the boundary conditions
$$\begin{aligned} f(x, y) &= 0 \text{ on } \Gamma_1 \\ f(x, y) &= 1 \text{ on } \Gamma_3 \\ f(x, y) &= s(x, y) \text{ on } \Gamma_2 \text{ and } \Gamma_4 \end{aligned} \tag{2}$$
where $s(x, y)$ is any function that maps points on the curves Γ_2 or Γ_4 to $[0, 1]$ and is strictly increasing in the direction from Γ_1 to Γ_3. In our computations, $s(x, y)$ restricted to a curve is the arc-length function scaled to $[0, 1]$. On the

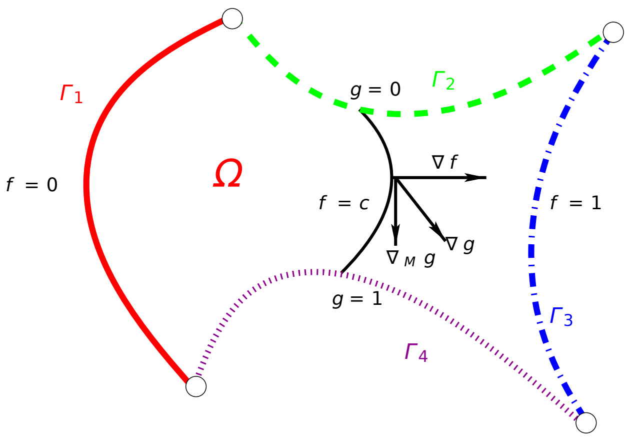


Fig. 2. Covariant gradient of g on a level set of f

other hand, we can choose Neumann boundary conditions, i.e., we can set $<\nabla f, n>= 0$ on Γ_2 or Γ_4 where n is the normal vector at corresponding point. However, we observed in our computations that this boundary conditions lead to unsatisfactory solutions.

Step 1.2. Once f is found, the scalar function g is generated such that g minimizes the harmonic energy on the level sets of f. More precisely, we define $M = f^{-1}(c) \cap \Omega$ for some constant c and we aim at solving the following minimization problem

$$E_2(g) = \int_M ||\nabla_M g||^2 \mathrm{d}volM \to \min_g$$

subject to the boundary conditions

$$g(x, y) = 0 \text{ if } (x, y) \in \Gamma_4$$
$$g(x, y) = 1 \text{ if } (x, y) \in \Gamma_2.$$

Note that there are only 2 points in the boundary conditions which are the intersections of M with 2 curves Γ_2 and Γ_4. The covariant gradient of g on M is the projection of the gradient of g onto the tangent line with the normal vector ∇f, see Figure 2. Its length can be calculated by

$$||\nabla_M g||^2 = ||\nabla g - \langle \nabla g, \nabla f\rangle \frac{\nabla f}{||\nabla f||^2}||^2.$$

Hence, we can rewrite the harmonic energy as

$$\begin{aligned} E_2(g) &= \int_M ||\nabla g - \langle \nabla g, \nabla f\rangle \frac{\nabla f}{||\nabla f||^2}||^2 \mathrm{d}volM \\ &= \int_\Omega ||\nabla g - \langle \nabla g, \nabla f\rangle \frac{\nabla f}{||\nabla f||^2}||^2 \delta(f - c)||\nabla f|| \mathrm{d}\mathbf{x} \qquad (3) \end{aligned}$$

where $\delta(.)$ is the Dirac delta function in the sense of distributions. In the last equality, we use the formula

$$\int_M \mathcal{R} \mathrm{d}volM = \int_\Omega \mathcal{R}\delta(f-c)||\nabla f||\mathrm{d}\mathbf{x}$$

where $\mathcal{R}$ is some function. Next, by means of minimizing simultaneously the harmonic energy for all level sets of f contained in Ω, we can eliminate the Dirac delta function in the Eq. (3) and formulate our minimization problem as

$$\int_\Omega ||\nabla g - \langle \nabla g, \nabla f\rangle \frac{\nabla f}{||\nabla f||^2}||^2 ||\nabla f|| \mathrm{d}\mathbf{x} \to \min_g \tag{4}$$

subject to the boundary conditions

$$\begin{aligned} g(x,y) &= 0 \text{ if } (x,y) \in \Gamma_4 \\ g(x,y) &= 1 \text{ if } (x,y) \in \Gamma_2. \end{aligned}$$

Note that we do not need to specify any boundary condition on the two curves Γ_1 and Γ_3.

4.2 Step 2: Finding a Spline Approximation S of F^{-1}

We use least squares fitting to find the mapping S that maps the unit square to $\mathbb{R}^2$. We write S in the tensor product B-spline form as

$$S(u,v) = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} M_{i,\mathcal{U}}(u)M_{j,\mathcal{V}}(v)\mathbf{d}_{ij} = B(u,v)\cdot D,\ (u,v)\in[0,1]^2,$$

where $D = (\mathbf{d}_{ij})$ is the vector of control points and $B(u,v) = M_{i,\mathcal{U}}(u)M_{j,\mathcal{V}}(v)$ is the vector of tensor product B-splines. We denote by $\mathcal{U}$ and $\mathcal{V}$ two given knot vectors with degree-fold boundary knots 0 and 1. The index sets $\mathcal{I}$ and $\mathcal{J}$ are determined by the knot sequences and degrees of the B-splines. Then, we formulate the problem to find $S \approx F^{-1}$ as the following least squares problem

$$\int_\Omega ||S\circ F - \mathrm{id}||^2 + R_f \to \min_D \tag{5}$$

where R_f is the fairness term

$$R_f = \omega_1 \int_{[0,1]^2} (S_u^2 + S_v^2)\mathrm{d}u\mathrm{d}v + \omega_2 \int_{[0,1]^2} (S_{uu}^2 + 2S_{uv}^2 + S_{vv}^2)\mathrm{d}u\mathrm{d}v$$

and ω_1 and ω_2 are weights. The least squares problem is a quadratic minimization problem with respect to D. Therefore, it is relatively simple to find the solution by solving a linear system.

5 Injectivity

Before going to the details of the implementation, we discuss the injectivity of the mapping F. The argument for three dimensional case is similar. We will consider each coordinate function of F separately.

The first coordinate function f minimizes the harmonic energy over the whole domain Ω. The Euler-Lagrange equation associated with (1) is the Laplace equation $\Delta f = 0$. In fact, if f is a solution of (1), f is a weak harmonic function. By the maximum principle, we conclude that each level set of f is a simple curve segment in Ω.

For the second coordinate function g, the Euler-Lagrange equation corresponding to the problem (4) is

$$\frac{1}{||\nabla f||}\nabla \cdot \left(\left(\nabla g - \langle \nabla g, \nabla f \rangle \frac{\nabla f}{||\nabla f||^2} \right) ||\nabla f|| \right) = 0 \tag{6}$$

So, if g is a solution of (4), then g is a weak solution of (6). Conversely, if g is a weak solution of (6), then the weak form of (6) also is satisfied on the manifold $M = f^{-1}(c) \cap \Omega$ where c is some constant. Therefore, g is a weak harmonic function on the manifold M and g satisfies the maximum principle on M.

Proposition 1. *If f is the solution of* (1) *and g is the solution of* (4)*, then the mapping $F = (f, g)^T$ from Ω to the unit square is injective.*

Proof. We assume that there are 2 points P_1 and P_2 on Ω such that $F(P_1) = F(P_2)$. So, P_1 and P_2 must lie on a level set of f. By the assumption, we have that $g(P_1) = g(P_2)$. Since each level set of f is a simple curve segment, and g is continuous on that level set and has no local maxima or minima, we conclude that P_1 and P_2 must lie on the boundary of that level set of f or more precisely on the intersection of that level set of f with boundary of Ω. However, the intersection set consists of only 2 points. Since g satisfies the boundary conditions, the values of g on those points are different. This is a contradiction. □

However, because we must solve the variational problems numerically, the discretization error may potentially destroy the injectivity. It depends on how well we can approximate the exact solutions. This will be described in the next section.

6 Implementation and Examples

In this section, we present how to solve the problems (1), (4) and (5). Then we present some experiments with our method.

6.1 Solving Variational Problems

To solve the variational problems (1) and (4) in the previous section, we employ a least squares meshless method using B-splines which is inspired by web-splines

[9]. We want to find approximate solutions $\hat{f} \approx f$ and $\hat{g} \approx g$ in the finite dimensional space spanned by B-splines. The reason to use this space is that B-spline functions are locally piecewise polynomial with rectangular supports and provide possibilities for refinement. Moreover, we can obtain a very good approximation while having only to use a few basis functions. So, once we obtain the solution $\hat{f}$, we can evaluate its values very fast. This is important because in the next variational problem for g, fast evaluation of $\hat{f}$ and its gradient speeds up the time for computing g.

For finding an approximate solution $\hat{f}$, we write

$$\hat{f} = \sum_{(i,j)\in\mathcal{I}'} N_{i,\mathcal{X}}(x)N_{j,\mathcal{Y}}(y)c_{ij}$$

where c_{ij} are real coefficients. The basis functions $N_{i,\mathcal{X}}$ and $N_{j,\mathcal{Y}}$ are B-splines of degree d_1 and d_2 with respect to knot sequences $\mathcal{X}$ and $\mathcal{Y}$ which are determined by the projections to the axes of the edges of the bounding box $\mathcal{B}$ of Ω. In our implementation, we use uniform knot vectors in x and y direction and consider only those tensor-product basis functions whose supports overlap the domain. By using the penalty method to enforce the boundary conditions, we arrive at the following least squares problem

$$\lambda \int_{\partial\Omega} (\hat{f} - f_{\mathcal{D}})^2 \mathrm{d}s + \int_{\Omega} ||\nabla \hat{f}||^2 \mathrm{d}\mathbf{x} \to \min_{c_{ij}} \tag{7}$$

where λ is a large weight, e.g., 10^3 and $f_{\mathcal{D}}$ is the function defined by Dirichlet boundary conditions of f on the boundary of Ω. The above minimization problem is quadratic with respect to c_{ij}, so it can be solved by solving a linear system. Note that we have to evaluate the second integral on the domain Ω which is unknown except for its boundary. At this point, an approximate implicitization algorithm, such as [14] or [6], comes to play its role. So, the boundary of Ω can be represented by the zero level set of an auxiliary function. The domain inside the boundary curve corresponds to the points on which the function possesses non-positive values. In other words, we can construct a characteristic function χ of the domain Ω. The second term in (7) can be rewritten as

$$\int_{\mathcal{B}} ||\nabla \hat{f}||^2 \chi(\Omega) \mathrm{d}\mathbf{x}$$

The linear system may be singular when the supports of some B-splines basis functions lie outside the domain Ω. Also, it may have a high condition number, if the supports have only a small intersections with the domain. In the web-spline method [9], this issue is dealt with by coupling some functions along the boundary. Instead of implementing the full web-spline method, we use a simpler approach. We simply set the value of χ to 1 on Ω and to a small value, e.g. 10^{-3}, outside Ω. We evaluate the integral by Gauss quadrature. The first integral term in (7) is evaluated by taking a weighted sum of finitely many points.

For finding the approximate solution $\hat{g}$, we use a similar method as above. The least squares problem is

$$\lambda \int_{\Gamma_2 \cup \Gamma_4} (\hat{g} - g_{\mathcal{D}})^2 \mathrm{d}s + \int_{\Omega} ||\nabla \hat{g} - \langle \nabla \hat{g}, \nabla \hat{f} \rangle \frac{\nabla \hat{f}}{||\nabla \hat{f}||^2} ||^2 ||\nabla \hat{f}|| \mathrm{d}\mathbf{x} \to \min_{c'_{ij}} \tag{8}$$

where $g_{\mathcal{D}}$ is the function defined by Dirichlet boundary conditions of g on the two curves Γ_2 and Γ_4. We denote c'_{ij} the new variables to indicate that we can use a different discretization space but the domain is the same. Analogously, the integral over Ω can be transformed to the integral over $\mathcal{B}$ by using the characteristic function. Again, this is a quadratic minimization problem with respect to the coefficients of $\hat{g}$.

6.2 Spline Approximation of the Inverse Mapping

Here we want to describe in detail how to solve the problem (5). The first integral term in (5) can be written as

$$\iint_{\mathbb{R}^2} ||B(f(x,y), g(x,y)) \cdot D - (x,y)^T||^2 \chi_{\Omega}(x,y) \mathrm{d}x \mathrm{d}y$$

where the characteristic function $\chi_{\Omega}(x,y)$ is already known from approximate implicitization but now it is set to 0 outside Ω. We evaluate the integral by numerical quadrature, i.e., taking uniform grid of points on the bounding box $\mathcal{B}$ of Ω. In order to obtain a good approximation for the boundary of Ω, we add more terms that are integrals over the boundary of Ω as

$$\omega_0 \Bigg(\int_{\Gamma_1} ||B(0, g(\mathbf{p})) \cdot D - \mathbf{p}||^2 \mathrm{d}\mathbf{p} + \int_{\Gamma_3} ||B(1, g(\mathbf{p})) \cdot D - \mathbf{p}||^2 \mathrm{d}\mathbf{p} \\ + \int_{\Gamma_2} ||B(f(\mathbf{p}), 1) \cdot D - \mathbf{p}||^2 \mathrm{d}\mathbf{p} + \int_{\Gamma_4} ||B(f(\mathbf{p}), 0) \cdot D - \mathbf{p}||^2 \mathrm{d}\mathbf{p} \Bigg)$$

where ω_0 is a large weight to indicate that we want to achieve a better approximation on the boundary. Once we have found the spline approximation, we use the L^∞ norm to estimate the error on the boundary of the domain. If the accuracy is not sufficient, then one may increase the numbers of degrees of freedom via knot insertion.

The obtained parametrization is an approximation on the inverse of the constructed coordinate functions, hence it does not reproduce the original boundary curves in general. It is possible to preserve the original boundaries, provided that the computed parameterization is compatible with the original parameterization of the boundaries. In this situation, the given control points of the boundary curves can be used as known coefficients in the fitting process.

The compatibility of the parametrizations can be achieved by composing the computed parameterization with two monotonic re-parameterizations, one for each coordinate function. Each of these monotonic re-parameterizations takes constant values 0 and 1 on two of the four boundaries, and it is determined by the original and the computed parametrizations on the other two boundaries. It can be extended to the entire domain by simple linear interpolation, which preserves the monotonicity along the parameter lines.

We did not (yet) implement this approach. We feel that the main advantage of IGA is that simulation and geometric design are based on the same geometry representation, and not necessarily in using an already existing CAD description. This will often be impossible, e.g., for domains possessing more or less than four B-spline boundary curves.

6.3 Putting Things Together

Now we summarize the implementation step by step. First, a domain defined by 4 curves is given. Next, we have to parameterize the curves Γ_2 and Γ_4 to $[0, 1]$ in order to define the boundary conditions (2). This is performed by arc length parameterization scaled to $[0, 1]$. Next, we find the bounding box of Ω. It is the domain for spline approximation of the functions f and g. Then, we generate the characteristic function for Ω by approximate implicitization. Finally, the two main steps, variational problems solving and spline approximation as described above, are performed.

6.4 Examples

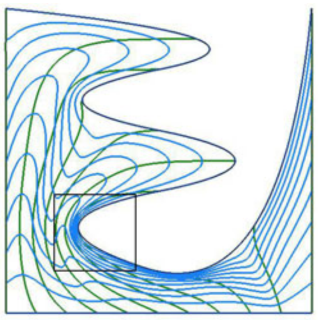
Our method is implemented in *C++* and uses *Gotools* [16] for B-splines manipulations. In addition, for solving the various arising sparse linear systems, we employ the *umfpack* routine [4]. The results in this paper are visualized using *OpenGL* and *Qt*. All examples in this paper are performed on a 2.8 GHz Intel Core 2 Duo CPU with 4 GByte Ram.

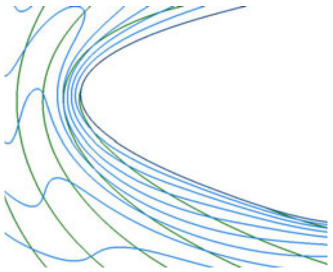
Example 1. Figure 3 demonstrates our method for a weird planar shape. Three boundary curves of that shape are line segments and the remain curve is modeled as quadratic B-splines curve with 7 control points. The picture (3a) shows the resulting spline approximation S and the level sets of f and g are approximated to the green curves and blue curves, respectively. The estimate errors in L^∞ norm are 0.002, 0.0004, 0.004 and 0.001 with respect to the boundary curve 1 to 4. We check the injectivity of F and S by evaluating its Jacobian at each sample point inside the domain. Then we plot the Jacobian of S in the picture (3c).

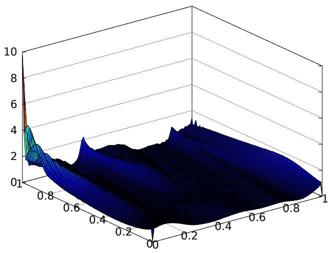
It is also possible to certify the injectivity of the parameterization. For example, sufficient conditions for injectivity are presented in the recent paper [18]. However, this method requires additional computational effort.

The construction of the parametrization map is not symmetric. If we swap the order for constructing f and g, we obtain the result plotted in the picture (3d). It is not a good parameterization, although F is still injective.

Example 2. The second example in Figure 4 is a shape representing the map of Austria. The four boundary curves of this shape are quadratic B-spline curves. The spline which parameterize the shape has 62×62 control points.


(a) The resulting parameterization


(b) Zoom in



(c) Jacobian of the resulting spline surface

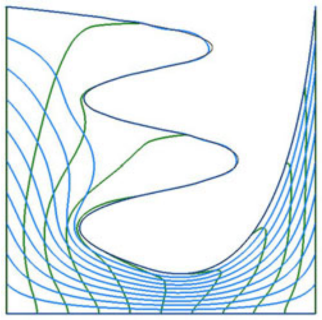

(d) The resulting parameterization in the reverse order

Fig. 3. Parameterization for a weird planar shape

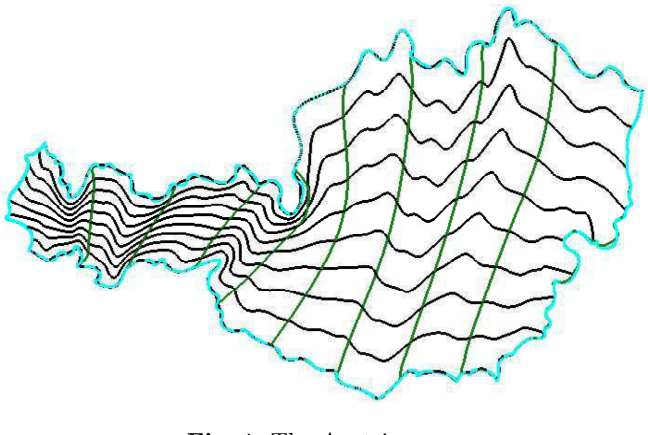
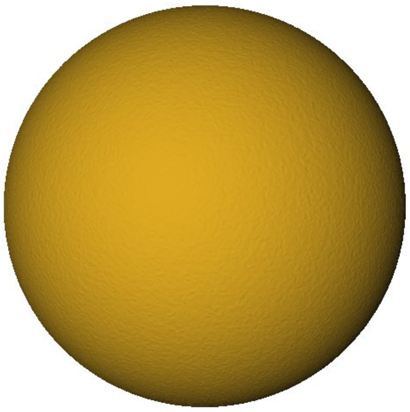
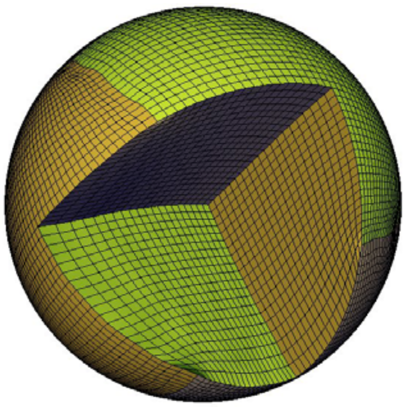
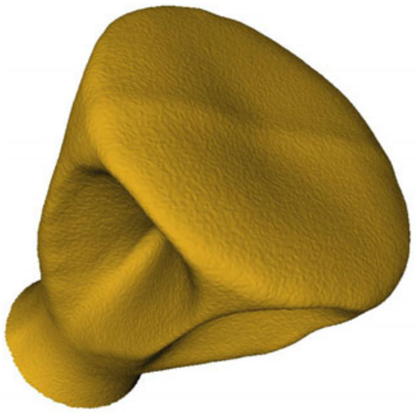


Fig. 4. The Austria map

Example 3. We shows some results for 3D case in Figure 5. In this situation, the input data is a closed triangular meshes. To be able to perform our the method, two preprocessing steps must be done. First, the mesh is segmented into 6 faces such that each face has exactly 4 incident faces. In other words, we should have the same topology as the unit cube. Second, we parameterize 4 faces to obtain boundary data for the variational problems. In order to visualize the quality of the parametrization, the figures (5b) and (5d) show a cut-off view of the interior of the solid objects.


(a) The triangular mesh model for a sphere


(b) The resulting parameterization for the sphere


(c) The triangular mesh model for a base of a screwdriver

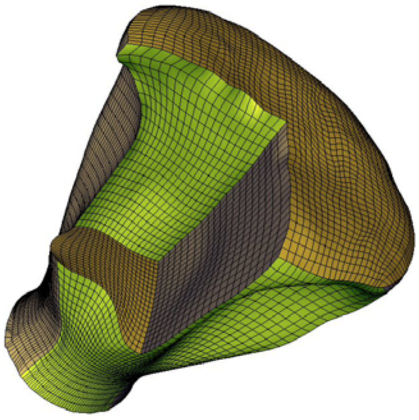

(d) The resulting parameterization for the base of the screwdriver

Fig. 5. Volumetric parameterization for two solids represented by closed triangular meshes

7 Conclusion

In this paper, we proposed a framework to compute an injective mapping from a domain defined by its boundary curves (surfaces) to the unit square (unit cube). This mapping is defined via solutions of two (three in 3D) variational problems. The final spline representation for the domain was constructed as an approximation of the inverse of the computed mapping. We also demonstrated that our method works efficiently for some complicated domains.

References

1. Aigner, M., Heinrich, C., Jüttler, B., Pilgerstorfer, E., Simeon, B., Vuong, A.V.: Swept volume parameterization for isogeometric analysis. In: Hancock, E.R., Martin, R.R., Sabin, M.A. (eds.) Mathematics of Surfaces XIII. LNCS, vol. 5654, pp. 19–44. Springer, Heidelberg (2009)
2. Bertalmío, M., Cheng, L.T., Osher, S., Sapiro, G.: Variational problems and partial differential equations on implicit surfaces. J. Comput. Phys. 174(2), 759–780 (2001)
3. Cohen, E., Martin, T., Kirby, R.M., Lyche, T., Riesenfeld, R.F.: Analysis-aware modeling: Understanding quality considerations in modeling for isogeometric analysis. Comput. Methods Appl. Mech. Engrg. (2009)
4. Davis, T.: http://www.cise.ufl.edu/research/sparse/umfpack/
5. Dinh, H.Q., Yezzi, A., Turk, G.: Texture transfer during shape transformation. ACM Trans. Graph. 24(2), 289–310 (2005)
6. Dokken, T., Thomassen, J.B.: Weak approximate implicitization. In: Shape Modeling and Applications, International Conference on Shape Modeling and Applications (SMI 2006), p. 31 (2006)
7. Farin, G., Hansford, D.: Discrete Coons patches. Comput. Aided Geom. Des. 16(7), 691–700 (1999)
8. Gu, X., Yau, S.T.: Global conformal surface parameterization. In: Proceedings of the First Eurographics Symposium on Geometry Processing, pp. 127–137. Eurographics Association (2003)
9. Höllig, K.: Finite element methods with B-splines. Frontiers in Applied Mathematics, vol. 26. Society for Industrial and Applied Mathematics, Philadelphia (2003)
10. Hormann, K., Polthier, K., Sheffer, A.: Mesh parameterization: Theory and practice. In: SIGGRAPH Asia 2008 Course Notes, Singapore, vol. (11). ACM Press, New York (2008)
11. Hoschek, J., Lasser, D.: Fundamentals of computer aided geometric design. Wellesley (1993)
12. Hughes, T.J.R., Cottrell, J.A., Bazilevs, Y.: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. Computer Methods in Applied Mechanics and Engineering 194(39-41), 4135–4195 (2005)
13. Jost, J.: Riemannian Geometry and Geometric Analysis. Springer, Heidelberg (2005)
14. Jüttler, B., Felis, A.: Least-squares fitting of algebraic surfaces. Advances in Computational Mathematics 17, 135–152 (2002)
15. Martin, T., Cohen, E., Kirby, R.M.: Volumetric parameterization and trivariate B-spline fitting using harmonic functions. Computer Aided Geometric Design 26(6), 648–664 (2009)

16. SINTEF, http://www.sintef.no/Projectweb/Geometry-Toolkits/GoTools/
17. Voruganti, H.K., Gupta, V., Dasgupta, B.: Domain mapping for spherical parametrization using harmonic function theory. In: IISc Centenary-International Conference on Advances in Mechanical Engineering, IC-ICAME (2008)
18. Xu, G., Mourrain, B., Duvigneau, R., Galligo, A.: Parametrization of computational domain in isogeometric analysis: methods and comparison. Computer Methods in Applied Mechanics and Engineering 200(23-24), 2021–2031 (2011)

Nonlinear L_1C^1 Interpolation: Application to Images

E. Nyiri, O. Gibaru, and Ph. Auquiert

Arts et Métiers ParisTech, L2MA,
8 bd Louis XIV, 59046 Lille Cedex, France
Eric.Nyiri@ensam.eu,
Olivier.Gibaru@ensam.eu
http://www.l2ma.fr

Abstract. In this article, we address the problem of interpolating data points lying on a regular grid by C^1-continuous L_1-bicubic spline surfaces. Our algorithm is based on a local univariate L_1 minimization method which enable us to calculate first derivative values for C^1-cubic spline curves. In order to construct the interpolation surface, we calculate four derivative values at each data point using this local method. At is was shown in [17], our local interpolation L_1 cubic spline curve algorithm preserves well the shape of the data even for abrupt changes.The sequential computational complexity of this local method is linear and the parallel computational complexity is O(1). Consequently, we can address in this manner data on large grids. In order to keep this linear complexity for spline surface interpolation, we define an interpolation scheme based on four linear directions so as to construct our L_1-bicubic surface. Some image interpolation examples show the efficiency of this non linear interpolation scheme.

Keywords: L_1 spline, interpolation, shape preserving, images.

Introduction

A common requirement in data interpolation is that curves or surfaces obtained "preserve shape", which means they express the geometric properties of the interpolated data in accordance with human perception. These geometric properties are variously interpreted as linearity, monotonicity, convexity and smoothness. Conventional 2D splines, which are calculated by minimizing the square of the L_2 norm of the second partial derivatives of a cubic piecewise polynomial interpolant, represent sufficiently "smooth" data quite well. However, they often have extraneous, nonphysical oscillations when used for interpolation of data with abrupt changes.

Recently, a new kind of splines called cubic L_1 splines has arisen, cf. [4,5,6,8,11,13]. Cubic L_1 splines, which are calculated by minimizing the L_1 norm of the second derivatives of a C^1-smooth piecewise cubic interpolant, "preserve the

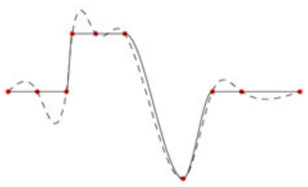


Fig. 1. L_2 (dotted line) versus L_1(solid line) global interpolations

shape" of data with abrupt changes (cf. [10] [12]) as we can see in Figure 1. In [17], we show that our local cubic L_1 spline interpolation algorithm preserves linear parts and does not produce Gibbs phenomena.

The advantageous properties of L1 splines in univariate situations suggest that L1 splines may have advantages in bivariate situations also, so we have decided to extend our local algorithm to bivariate interpolation for images. This bicubic model can be used for image applications like resampling [19,24,7] or warping [18].

The L_1 splines are generated by minimizing a nonlinear functional, cf. [10, 14,15]. Since nonlinear programming procedures for minimizing the L_1 spline functional are not yet practical for global data interpolation, a discretization of this functional is commonly used. Minimization of the discretized L_1 spline functional which is a nonsmooth convex programming problem, leads to solving of an overdetermined linear system that can be reduced to a linear program for which many methods are available(cf. [16], [23], [20], [21], [22]). The drawback of this global minimization is the calculation time which can be very important for big data sets.

We introduce a local univariate minimization method based on a sliding window defined over five points so as to smoothly interpolate data points. This local method allows us to define a fast computational algorithm based on analytically minimizing the L_1 spline functional over only five data points. Furthermore, we showed in [2] that a five-point window allows us to preserve the linear parts of the data points while in general the global method does not satisfy this property.

This paper is organized as follows. In Section 1, we give some results concerning C^1-continuous cubic L_1 spline interpolation on five points. Based on these results, we have defined a new interpolation strategy with a sliding five points window to create a local L_1C^1 interpolating method. In Section 2, we use this univariate method so as to construct bicubic spline surfaces. By applying our L_1C^1 interpolating algorithm to regular data grids in four directions, we generate a set of derivative values at each data point. We show the efficiency of this method by applying continuous transformations on some images.

1 Local L_1 Cubic Spline Minimization

The C^1 interpolating cubic spline curve is calculated by minimizing the L_1-norm of the second derivative vector of the spline. Let $a = u_1 < u_2 < \cdots < u_n = b$ be an arbitrary and strictly monotonic partition of the finite real interval $[a, b]$. If we denote by Δ the classical forward difference operator, we showed in [1] that the solution to this problem is obtained by minimizing the following functional

$$E(T_1, \ldots, T_n) = \sum_{i=1}^{n-1} \int_{\frac{-1}{2}}^{\frac{1}{2}} \|\Delta T_i + 6t(T_{i+1} + T_i - 2\frac{\Delta P_i}{\Delta u_i})\|_1 dt \tag{1}$$

where the $T_i \in \mathbb{R}^d$ are the first order derivative vectors at points P_i for $i = 1, \ldots, n$. As $E(T_1, \ldots, T_n)$ is not strictly convex, then its minima are not necessarily unique. To reduce the set of solutions, Lavery in [10] added a "regularization" term so as to select the derivative vectors T_i which are as short as possible in the L_1-norm. Consequently, a global C^1-continuous cubic L_1 spline is obtained by minimizing the following functional

$$E(T_1, \ldots, T_n) + \varepsilon \sum_{i=1}^{n} |T_i|, \tag{2}$$

where ε is a strictly positive real. As this problem is also nonlinear, this functional is discretized by using the midpoint rule method for each integral. The resulting problem raised by the L_1-minimization of linear system is solved by the Vanderbei, Meketon and Freedman primal affine algorithm defined in [23] and outlined in [12]. This algorithm was improved in [25] in order to reduce the calculation time. Nevertheless for big data sets, global calculus is prohibited. Thanks to this fact, we have calculated the exact solutions to this minimization problem (1) on a set of five data points. To do so, let us study the three-point case.

1.1 Univariate Cubic L_1C^1 Interpolation over Three Points

The following lemma gives the exact solution to the minimization of (1) with $n = 3$.

Let $(u_i, z_i)_{i=1,2,3}$ be three couples of real values where $u_1 < u_2 < u_3$ and the slopes be defined by $h_i = \frac{\Delta z_i}{\Delta u_i}$ for $i = 1, 2$. The solution of the minimization problem

$$\varphi_{h_1,h_2}(x) = \min_{(b_2,b_3)\in\mathbb{R}^2} \Phi(b_1, b_2, b_3) \tag{3}$$

with

$$\Phi(b_1, b_2, b_3) = \int_{\frac{-1}{2}}^{\frac{1}{2}} |\Delta b_1 + 6t(b_2 + b_1 - 2h_1)| dt + \int_{\frac{-1}{2}}^{\frac{1}{2}} |\Delta b_2 + 6t(b_3 + b_2 - 2h_2)| dt, \tag{4}$$

where b_1, b_2 and b_3 are the first derivative values at the three points is

$$\begin{array}{l} \text{a) if } b_1 \text{ is comprised between } h_1 + \frac{\sqrt{10}+1}{3}\left(h_2 - h_1\right) \text{ and } h_1 \text{ then} \\ b_2 = h_1 + \frac{\sqrt{10}-1}{3}\left(b_1 - h_1\right), b_3 = h_2 + \frac{\sqrt{10}-5}{5}\left(h_1 - h_2\right) + \frac{5-2\sqrt{10}}{5}\left(b_1 - h_1\right), \\ \\ \text{b) if } b_1 \text{ is comprised between } h_1 \text{ and } h_1 + \frac{\sqrt{10}-5}{5}\left(h_2 - h_1\right) \text{ then} \\ b_2 = h_1 - \frac{5+\sqrt{10}}{3}\left(b_1 - h_1\right),\ b_3 = h_2 + \frac{\sqrt{10}-5}{5}\left(h_1 - h_2\right) + \left(b_1 - h_1\right), \\ \\ \text{c) otherwise } b_2 = b_3 = h_2. \end{array} \tag{5}$$

This was demonstrated in [17].

For the five point algorithm defined hereafter, we need to calculate the subdifferential of this continuous convex function. Let us define the following functions:

$$\begin{array}{l} \varphi^1_{h_1,h_2}\left(x\right) = -\frac{3}{2}\left(x + h_2 - h_1\right) - \frac{\left(h_2-h_1-x\right)^2}{6\left(x+h_2-h_1\right)}, \\ \varphi^2_{h_1,h_2}\left(x\right) = \frac{8-4\sqrt{10}}{3}x + \frac{2\left(\sqrt{10}-1\right)}{3}\left(h_1 - h_2\right),\ \varphi^3_{h_1,h_2}\left(x\right) = \frac{2\left(\sqrt{10}-1\right)}{3}\left(h_1 - h_2\right), \\ \varphi^4_{h_1,h_2}\left(x\right) = \varphi^1_{h_1,h_2}\left(x\right),\ \varphi^5_{h_1,h_2}\left(x\right) = x - h_2 + h_1,\ \varphi^6_{h_1,h_2}\left(x\right) = -\varphi^1_{h_1,h_2}\left(x\right). \end{array}$$

Consequently, we can infer that

$$\varphi_{h_1,h_2}\left(x\right) = \sigma\varphi^k_{h_1,h_2}\left(x\right) \text{ if } x \in \left[\min\left(\sigma x^{k-1}_{h_1,h_2}, \sigma x^k_{h_1,h_2}\right), \max\left(\sigma x^{k-1}_{h_1,h_2}, \sigma x^k_{h_1,h_2}\right)\right], \tag{6}$$

where $\sigma = \{1 \text{ if } h_2 - h_1 \geq 0 \text{ and } -1 \text{ otherwise}\}$ and

$$\begin{cases} x^0_{h_1,h_2} = -\infty, x^1_{h_1,h_2} = \frac{\sqrt{10}+1}{3}\left(h_2 - h_1\right), x^2_{h_1,h_2} = 0, x^3_{h_1,h_2} = \frac{\sqrt{10}-5}{5}\left(h_2 - h_1\right), \\ x^4_{h_1,h_2} = -\frac{1}{2}\left(h_2 - h_1\right), x^5_{h_1,h_2} = -2\left(h_2 - h_1\right), x^6_{h_1,h_2} = +\infty. \end{cases} \tag{7}$$

1.2 Univariate Cubic L_1C^1 Interpolation over Five Points

Let $(u_i, z_i)_{i=1,\ldots,5}$ be five couples of real values where $u_1 < \cdots < u_5$ and the slopes be defined by $h_i = \frac{\Delta z_i}{\Delta u_i}$ for $i = 1, \ldots, 4$. Hence, the univariate L_1C^1 spline solution is obtained from

$$\min_{(b_1,\ldots,b_5)\in\mathbb{R}^5} \sum_{i=1}^{4} \int_{\frac{-1}{2}}^{\frac{1}{2}} \left|\Delta b_i + 6t(b_{i+1} + b_i - 2h_i)\right| dt$$

where the b_i are the derivative values of the spline at u_i. This functional is the sum of positive and convex continuous functions. It can be written by

$$\begin{array}{l} \min_{b_3\in\mathbb{R}}\left(\min_{(b_2,b_1)\in\mathbb{R}^2} \Phi(b_3, b_2, b_1) + \min_{(b_4,b_5)\in\mathbb{R}^2} \Phi(b_3, b_4, b_5)\right) \\ = \min_{b_3\in\mathbb{R}}\left(\varphi_{h_2,h_1}\left(b_3 - h_2\right) + \varphi_{h_3,h_4}\left(b_3 - h_3\right)\right) \end{array} \tag{8}$$

where Φ is defined by (4). Let us denote by $\partial\varphi_{h_i,h_j}(x)$ the subdifferential of $\varphi_{h_i,h_j}(x)$ at x (cf. [3], [9]). Since (8) is convex and continuous, its subdifferential is compact and nonempty.

We define $d_g(x) = \min \partial\varphi_{h_2,h_1}(x - h_2) + \min \partial\varphi_{h_3,h_4}(x - h_3)$ and $d_d(x) = \max \partial\varphi_{h_2,h_1}(x - h_2) + \max \partial\varphi_{h_3,h_4}(x - h_3)$ respectively the left and right derivative values of (8) at x. As function (8) is convex the minimal value is obtained for any b_3 such that $d_g(b_3).d_d(b_3) \leq 0$.

Consequently, the algorithm needed to define b_3 is

- create $\{\beta_k\}_{k=1\ldots10} =$ sorted list from the abscissa $\left(x^j_{h_3,h_4} + h_3, x^j_{h_2,h_1} + h_2\right)^5_{j=1}$
- calculate $\alpha_1 = \min_{k\in\{1,\ldots,10\}} (\beta_k$ such that $d_g(\beta_k)d_d(\beta_k) \leq 0$ or $d_d(\beta_k)d_g(\beta_k + 1) < 0)$
- calculate $\alpha_2 = \max_{k\in\{1,\ldots,10\}} (\beta_k$ such that $d_g(\beta_k)d_d(\beta_k) \leq 0$ or $d_d(\beta_k - 1)d_g(\beta_k) < 0)$.
- if $\alpha_1 = \alpha_2$ then $b_3 = \alpha_1 = \alpha_2$.
- if $\alpha_1 \neq \alpha_2$ and $d_d(\alpha_1) = 0$ and $d_g(\alpha_2) = 0$ then $b_3 = \min_{x\in[\alpha_1,\alpha_2]} \left|x - \frac{h2+h3}{2}\right|$ *(this solution preserves linear parts when possible).*
- if $\alpha_1 \neq \alpha_2$ and ($d_d(\alpha_1) \neq 0$ or $d_g(\alpha_2) \neq 0$) then b_3 is calculated by using a dichotomic search algorithm in $]\alpha_1, \alpha_2[$.

Since b_3 is obtained, we can calculate b_1, b_2, b_4 and b_5 by using Lemma 1.

1.3 Local L_1 Cubic Interpolation Spline

We define a five-point sliding window on a set of points and we calculate the derivative vector only for the middle point by using the previous algorithm (cf. Fig. 2). By translating the window, point by point over all the data, we obtain a derivative vector at each interpolation point. Hence, we are able to construct a cubic spline which is our local L_1 interpolation spline curve.

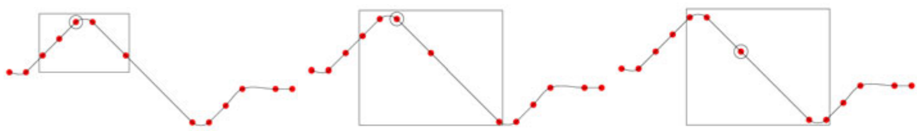


Fig. 2. Sliding window over the sets of points

We have shown in [17] that the minimization problem over five points retains linearity when three points are aligned. Hence, our local algorithm ensures that linear parts of the data will be interpolated by linear segments everywhere.

The previous algebraic algorithm over five points needs few operations and the total calculation time depends linearly of the number of interpolated data.

Another advantage of our local minimization method is that each window calculus can be done separately. So, the algorithm can be massively parallelized onto GPU cards or parallel computers.

These two above properties led us to extend this method on large data sets like images.

2 L_1 Bicubic Interpolation Method over Images

2.1 Bicubic Spline Surface Construction

A image can be defined as scattered data by using pixel values[1]. Applying transformations like rotations, resampling or warping to this discrete data set may produced blur or flattening. To overcome these problems, we can construct a continuous model by using a bicubic spline surface which interpolates the pixel values. In order to use our local L_1 minimization method for images, we have to extend our univariate algorithm to the bivariate case.

Firstly, we apply our local univariate algorithm onto each horizontal and vertical line of the image grid. This gives the first partial derivatives at each data point. Secondly, we calculate partial cross derivative values by using the diagonal directions of the image. The main advantage of this solution is that the two cross directions are also defined on a regular grid (see Fig. 3). Thirdly, by using these values, we can construct a grid of control points for a bicubic spline.

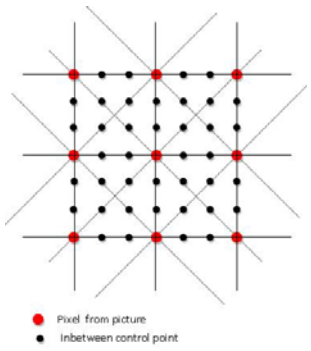


Fig. 3. Definition of control polygon for bicubic spline surface

2.2 Image Interpolation Results

This section shows that our algorithm has been tested with different kinds of images.

[1] These values are the components of the image so we can use RVB, YUV or other encoding systems.

As we have already mentioned, our univariate algorithm preserves linear parts. By using our surface construction algorithm, the part of the bicubic surface defined other a set of pixel values lying on a plane belong to this plane. In Figure 4.a, we have created three polygonal colored areas from black to white. As we can see in Fig 4, our method well preserves the different planes of data. Consequently, the color is uniform and there are no unwanted effects or oscillations near the edges. We have also tested our new algorithm with drawing (Fig. 5) and natural images. The so obtained interpolation surfaces can be used to transform these images.

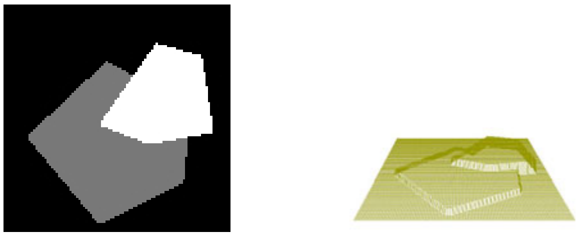
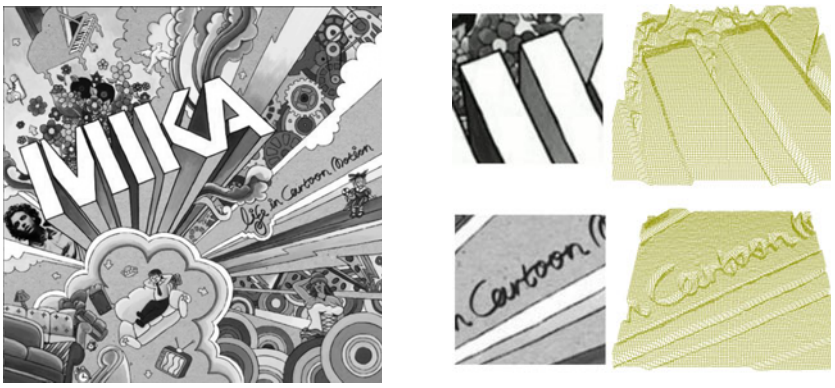

a.Three polygonal areas b.The associate bicubic spline surface

Fig. 4. Planar areas



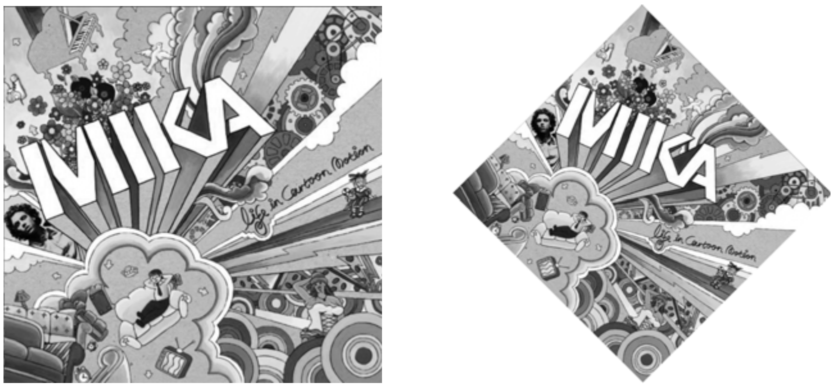
Drawing image

Fig. 5. Original, areas and their interpolation surfaces

2.3 Image Transformations

This section will not provide new algorithms for rotation, resampling, warping or other transformation. We apply continuous mathematical transformations on our continuous local L_1-bicubic interpolation spline surface in order to simply modify a image.

Rotation. Our method allows us to apply rotational transformation into the parameter-space of the bicubic spline interpolation model of an image. This method works well even for a rotation of 45 degrees which generally produces a bad result.



Original Rotated

Fig. 6. 45 rotated image

Resampling Application. The (re)sampling from a continuous model is not a problem contrarily to case of discrete data. This could be easily done whereas the resampling coefficient is not an integer. In Example 7, the resampling images are obtained with a scale coefficient of 1.8. We have applied one octave decimation (:2) onto the Mika image (cf. Figure 5) and after one octave super resolution (x2) in order to produce a test image. The calculated PSNR between the original and test image is 24,42 dB which is not so bad (cf. [24])

Warping Applications. The warping examples below had been obtained by applying continuous transformations into the parameter-space of the bicubic spline surfaces. Since the warping-model is continuous, it is very easy to apply to interpolation bicubic surface.

2.4 GPU Parallelization Results

Our algorithm complexity is $O(n)$ because it only depends on the number of data points even for the bivariate case. Since we use algebraic solution for each local univariate calculus, the calculation time is drastically decreased comparing to a global L1 surface algorithm ([13]). In addition, all five point calculus can be made independently because our algorithm use sliding windows. Thus, the calculation time can be optimized by using parallel computer architectures like GPU cards. If we compare CPU L1 interpolation algorithm to a GPU parallelized one[2] (see Fig. 9), we can see that the calculation time decreases. We can infer that as much as cores we have, as much as the global calculus time is decreasing; the limit is the time spent for one five point window calculus.

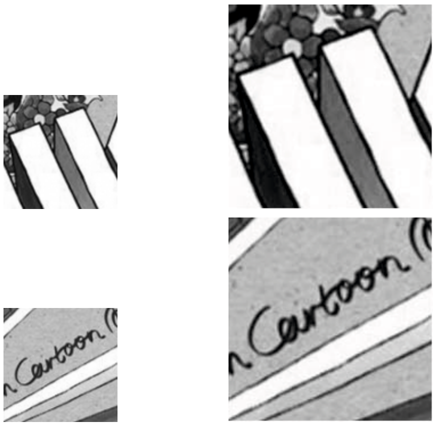


Fig. 7. Resampling

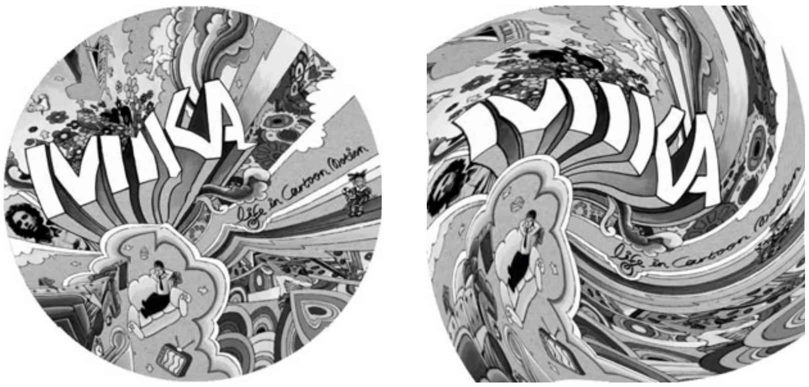


Fig. 8. Various warping images

[2] We used a NVidia Tesla Card with 240 TPU cores.

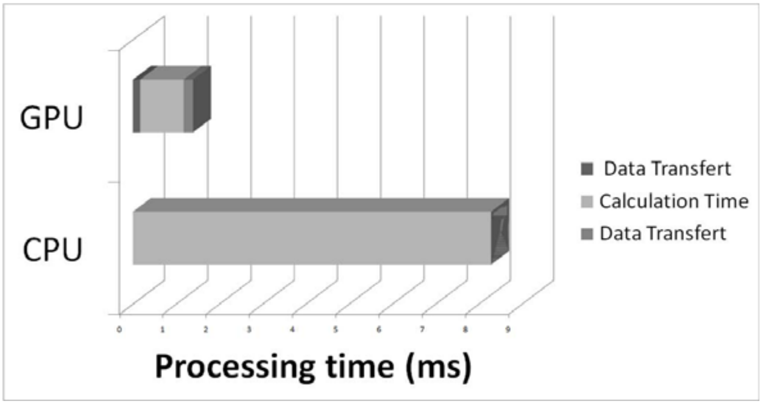


Fig. 9. Processing times for L1 curve interpolation over 3729 data point set

3 Conclusion

We have developed a L_1 bicubic interpolation surface algorithm which can be used over data grid and therefore with images. Because we're using a L1-norm minimization algorithm with a five point sliding window, we have got a method which has very good properties:

– shape preserving
– no Gibbs effects near abrupt data changes
– $O(n)$ complexity in sequencial calculus mode and $O(1)$ complexity in parallele calculus mode (with n = number of bivariate data points).
– massively parallelized possibility

After some tests, we have proved that the L_1 bicubic spline interpolation algorithm can be a good choice if someone want to create a continuous model from discrete data sets. Our local L_1-bicubic spline surface algorithm can be easily used to transform images. In a future work we will study the unstructured data set case.

Acknowledgments. The authors thank the referee for his helpful suggestions and comments.

References

1. Auquiert, P., Gibaru, O., Nyiri, E.: C^1 and C^2-continuous polynomial parametric L_p splines ($p \geq 1$), Comp. Aided Geom. Design 24, 373–394 (2007)
2. Auquiert, P., Gibaru, O., Nyiri, E.: On the cubic L_1 spline interpolant to the Heaviside function. Numer. Algor. 46, 321–332 (2007)
3. Azé, D.: Eléments d'analyse convexe et variationnelle, Ellipses (1997)

4. Cheng, H., Fang, S.-C., Lavery, J.E.: Univariate cubic L_1 splines—A geometric programming approach. Math. Methods Oper. Res. 56, 197–229 (2002)
5. Cheng, H., Fang, S.-C., Lavery, J.E.: An efficient algorithm for generating univariate cubic L_1 splines. Comput. Optim. Appl. 29, 219–253 (2004)
6. Cheng, H., Fang, S.-C., Lavery, J.E.: Shape-preserving properties of univariate cubic L_1 splines. J. Comput. Appl. Math. 174, 361–382 (2005)
7. Chuah, C.-S., Leou, J.-J.: An adaptive image interpolation algorithm for image/video processing. Pattern Recognition 34, 2383–2393 (2001); Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan 621, Republic of China Received November 20, 1998; accepted October 24, 2000
8. Gilsinn, D.E., Lavery, J.E.: Shape-preserving, multiscale fitting of bivariate data by cubic L_1 smoothing splines. In: Chui, C.K., Schumaker, L.L., Stöckler, J. (eds.) Approximation Theory X:Wavelets, Splines, and Applications, pp. 283–293. Vanderbilt University Press, Nashville (2002)
9. Hiriart-Urruty, J.B., Lemaréchal, C.: Fundamentals of Convex Analysis. Springer, Heidelberg (2001)
10. Lavery, J.E.: Univariate cubic L_p splines and shape-preserving, multiscale interpolation by univariate cubic L_1 splines. Comput. Aided Geom. Design 17, 319–336 (2000)
11. Lavery, J.E.: Shape-preserving, multiscale fitting of univariate data by cubic L_1 smoothing splines. Comput. Aided Geom. Design 17, 715–727 (2000)
12. Lavery, J.E.: Shape-preserving, multiscale interpolation by bi- and multivariate cubic L_1 splines. Comput. Aided Geom. Design 18, 321–343 (2001)
13. Lavery, J.E.: The state of the art in shape preserving, multiscale modeling by L_1 splines. In: Lucian, M.L., Neamtu, M. (eds.) Proceedings of SIAM Conference on Geometric Design Computing, pp. 365–376. Nashboro Press, Brentwood (2004)
14. Lavery, J.E.: Shape-preserving, first-derivative-based parametric and non parametric cubic L_1 spline curves. Comput. Aided Geom. Design 23, 276–296 (2006)
15. Lavery, J.E.: Shape-preserving univariate cubic and higher-degree L_1 splines with function-value-based and multistep minimization principles. Computer Aided Geom. Design 26, 1–16 (2009)
16. Lustig, I.J., Marsten, R.E., Shanno, D.F.: Interior point methods for linear programming: computational state of the art. ORSA J. Comput. 6, 1–14 (1994)
17. Nyiri, E., Gibaru, O., Auquiert, P.: Fast $L_1^kC^k$ polynomial spline interpolation algorithm with shape-preserving properties. Comp. Aided Geom. Design 28, 65–74 (2010/2011)
18. Niu, Y., Liu, F., Li, X., Gleicher, M.: Image resizing via non-homogeneous warping. Multimed Tools Appl. (2010); Published Online
19. Unser, M., Aldroubi, A., Eden, M.: Fast-B-SplineTransforms for Continuous Image Representation and Interpolation. IEEE Transsactions on Pattern Analysis and Machine Intelligence 13(3), 277–285 (1991)
20. Vanderbei, R.J.: Affine-scaling for linear programs with free variables. Math. Program. 43, 31–44 (1989)
21. Vanderbei, R.J.: LOQO An interior point code for quadratic programming, Statistics and Operations Research Technical Report SOR-94-15, Princeton University (1995)
22. Vanderbei, R.J.: Linear Programming: Foundations and Extensions, 2nd edn. Kluwer Academic, Boston (2001)

23. Vanderbei, R.J., Meketon, M.J., Freedman, B.A.: A modification of Karmarkar's linear programming algorithm. Algorithmica 1, 395–407 (1986)
24. van Ouwerkerk, J.D.: Image super-resolution survey. Image and Vision Computing 24, 1039–1052 (2006)
25. Wang, Y., Fang, S.-C., Lavery, J.E.: A compressed primal-dual method for generating bivariate cubic L_1 splines. Journal of Computational and Applied Mathematics 201, 69–87 (2007)

Normal Multi-scale Transforms for Surfaces

Peter Oswald[⋆]

Jacobs University Bremen, School of Engineering and Science,
Campus Ring 1, 28759 Bremen, Germany
p.oswald@jacobs-university.de
http://www.faculty.jacobs-university.de/poswald

Abstract. We prove well-posedness, convergence, and detail decay estimates for the normal triangular mesh multi-scale transform for $C^{1,\alpha}$ graph surfaces given in the simplest case when the subdivision rule S used for base point prediction is given by edge midpoint insertion. A restrictive assumption is that the initial triangular mesh needs to be quasi-regular and of small enough mesh-size. We also provide numerical evidence with other S for dyadic refinement (Butterfly, Loop), and propose a modification of the normal scheme resulting in improved detail decay for smooth surfaces.

Keywords: Nonlinear geometric multi-scale transforms, linear subdivision, surface representation, detail decay.

1 Introduction

Normal multi-scale transforms (MTs) for curves and surfaces [7] are nonlinear MTs based on a linear subdivision operator S and a nonlinear transformation for the detail part involving approximate normal directions. They have been used for multi-scale representation and compression of geometric objects [12,14,4], for adaptive approximation of level curves [2], in image analysis [1], and recently for interface tracking [17]. Roughly speaking, normal MT starts from an initial triangular mesh on Σ (a curve in $\mathbb{R}^2$ or a surface in $\mathbb{R}^3$) represented by its vertex set $\mathbf{v}^0$, and creates denser triangular meshes with vertex sets $\mathbf{v}^1, \mathbf{v}^2, \ldots$ on Σ by recursively repeating the following steps:

– Given $\mathbf{v}^{j-1} \subset \Sigma$ and a primal linear subdivision scheme S for triangular meshes, a set $\hat{\mathbf{v}}^j = S\mathbf{v}^{j-1}$ of base points is computed, these points typically do not belong to Σ.
– Similarly, a set $\hat{\mathbf{n}}^j$ of approximate normals of unit length (one for each base point) is computed, e.g., by taking suitable averages of normal directions to the triangles in the mesh associated with $\mathbf{v}^{j-1}$.
– By finding the intersection points $\mathbf{v}_i^j$ of the family of parametric "normal" lines $\hat{\mathbf{v}}_i^j + s\hat{\mathbf{n}}_i^j$ passing through the base points from $\hat{\mathbf{v}}^j$ in the direction given by the corresponding approximate normal from $\hat{\mathbf{n}}^j$ with Σ, one arrives

[⋆] Work supported by DFG grant OS-122/3-1.

at the new triangular mesh with vertex set $\mathbf{v}^j$ on Σ (with the topology inherited from the basepoint mesh), and the sequence d^j of scalar details d_i^j (given by the projection of $\mathbf{v}_i^j - \hat{\mathbf{v}}_i^j$ onto $\hat{\mathbf{n}}_i^j$) to be stored. This task is the most ambiguous one, as no or many intersection points with Σ may exist resp. mesh obstructions may occur. The vertex set $\mathbf{v}^j$ can be recovered from $\mathbf{v}^{j-1}$ and d^j by first computing $\hat{\mathbf{v}}^j$, $\hat{\mathbf{n}}^j$ from $\mathbf{v}^{j-1}$, and then using the reconstruction formula

$$\mathbf{v}^j = \hat{\mathbf{v}}^j + d^j \hat{\mathbf{n}}^j,$$

where the multiplication $d^j \hat{\mathbf{n}}^j$ has to be executed entry-wise.

If, with proper specification of the intersection procedure and under certain conditions on $\mathbf{v}^0$, this construction does not fail (well-posedness), we obtain the normal MT $\Sigma \to \{\mathbf{v}^0, d^1, d^2, \ldots\}$ and can recover Σ as limit of triangular surfaces by computing $\mathbf{v}^j$, $j \geq 1$, using the reconstruction formula recursively.

Normal MTs for smooth curves in $\mathbb{R}^2$ have been investigated in [3,16,10], and are well-understood by now. In this note, we make a first step towards a theory of normal MTs for smooth surfaces in $\mathbb{R}^3$ by investigating the normal MT for $C^{1,\alpha}$ surfaces $\Sigma \subset \mathbb{R}^3$ with the simplest S (based on edge midpoint insertion) for triangular meshes. The definition and basic properties of this transform and the underlying triangular meshes will be introduced in Section 2. Our main result concerning the well-posedness, convergence, and guaranteed detail decay of order $\mathrm{O}(2^{-j(1+\alpha)})$ of this MT for $C^{1,\alpha}$ graph surfaces, and sufficiently dense and regular initial meshes will be stated and proved in Section 3. Section 4 discusses the possible extensions of this initial result to more general S, and provides numerical evidence on detail decay for smooth surfaces. In particular, we show that the influence of extraordinary vertices can practically be neglected. We also introduce a new normal multi-scale transform with improved detail decay, consisting of a clever combination of Loop subdivision (for predicting base points and approximate normals) and Butterfly subdivision (for predicting a point closer to the surface along the normal line).

2 Definitions and Preliminary Facts

2.1 Surfaces and Triangular Meshes

For the study of normal MTs for closed 2-dimensional $C^{1,\alpha}$ manifolds Σ embedded in $\mathbb{R}^3$ (the treatment of boundaries is beyond the scope of our investigation), due to the locality of the used subdivision rule and the density assumptions on the initial meshes $\mathbf{v}^0$ necessary for our asymptotic analysis, we may resort to graph surfaces $\Sigma = \{\mathbf{v}_P := (P, f(P)) : \ P := (x, y) \in \mathbb{R}^2\} \subset \mathbb{R}^3$ given by a $C^{1,\alpha}$ function $f : \mathbb{R}^2 \to \mathbb{R}$ with globally bounded gradient and Lipschitz constants, i.e., there are constants C_0, C_1 such that

$$|\mathbf{n}_P| = \sqrt{1 + |\nabla f(P)|^2} \leq C_0,$$

where $\mathbf{n}_P = (-\nabla f(P), 1)$ is the normal to the tangent plane to Σ at $\mathbf{v}_P$,

$$|\nabla f(Q) - \nabla f(P)| \leq C_1 |Q - P|^\alpha, \quad P, Q \in \mathbb{R}^2,$$

and consequently also

$$f(Q) = f(P) + \nabla f(P)(Q - P) + R(P, Q), \qquad |R(P, Q)| \leq C_1 |Q - P|^{1+\alpha},$$

where $P, Q \in \mathbb{R}^2$. Everywhere, $|\cdot|$ denotes the Euclidean norm in $\mathbb{R}^d$ with $d = 1, 2, 3$, depending on context. Products of vectors in $\mathbb{R}^d$, if not specified otherwise, are to be understood as scalar products.

We consider triangular meshes on Σ generated by triangulations in $\mathbb{R}^2$, i.e., if $\mathcal{V}$ is the vertex set of a triangulation $\mathcal{T}$ in $\mathbb{R}^2$ then $\mathbf{v} := \{\mathbf{v}_P : P \in \mathcal{V}\}$ represents the vertex set of a triangular mesh on Σ, with the mesh topology (edges, faces) inherited from $\mathcal{T}$. The edges $\mathbf{e} = \mathbf{e}_{PQ} = \mathbf{v}_Q - \mathbf{v}_P$ can be identified with pairs neighboring vertices P and Q of $\mathcal{T}$. The global mesh-width $h(\mathbf{v})$ is defined as the supremum of all edge lengths $|\mathbf{e}|$. By $h_\mathcal{T}$ we denote the similarly defined mesh-width of $\mathcal{T}$. Since

$$|Q - P| \leq |\mathbf{e}| = |\mathbf{v}_Q - \mathbf{v}_P| \leq C_0 |Q - P| \tag{1}$$

we always have $h_\mathcal{T} \leq h(\mathbf{v}) \leq C_0 h_\mathcal{T}$. We call a triangular mesh with vertex set $\mathbf{v}$ regular with regularity constant $c = \sin\gamma$, $\gamma \in (0, \pi/3]$ (or in short c-regular), if the interior angles of any triangular face of the mesh are $\geq \gamma$. More conveniently, this can be expressed by requiring the inequality

$$|\mathbf{e} \times \tilde{\mathbf{e}}| \geq c |\mathbf{e}| |\tilde{\mathbf{e}}|$$

to hold for any two edges of any triangular face of the mesh. Without detailed proof, we state the following elementary properties.

Proposition 1. *Let Σ be a $C^{1,\alpha}$ graph surface, and $\mathbf{v}$ the vertex set of a c-regular triangular mesh on Σ, as described above.*
a) For any two edges $\mathbf{e}$ and $\tilde{\mathbf{e}}$ belonging to the same face, we have $|\mathbf{e}|/|\tilde{\mathbf{e}}|/ \leq 1/c$.
b) There is a $\bar{h}_1 > 0$ (depending on c, C_0 only) such that for $h(\mathbf{v}) \leq \bar{h}_1$, the underlying triangulation $\mathcal{T}$ of $\mathbb{R}^2$ is $\bar{c}$-regular, with some other $\bar{c} \geq c/(2C_0)$. Conversely, if $\mathcal{T}$ is $\bar{c}$-regular and $h_\mathcal{T}$ is small enough, then the associated triangular mesh on Σ is c-regular for some $c > \bar{c}/(2C_0^2)$.

Since $|\mathbf{e} \times \tilde{\mathbf{e}}| = |\mathbf{e} \times \bar{\mathbf{e}}| = |\tilde{\mathbf{e}} \times \bar{\mathbf{e}}|$, where $\bar{\mathbf{e}}$ denotes the remaining edge of the triangular face under consideration, we have

$$\frac{|\mathbf{e}|}{|\tilde{\mathbf{e}}|} = \frac{|\mathbf{e}||\bar{\mathbf{e}}|}{|\tilde{\mathbf{e}}||\bar{\mathbf{e}}|} \leq \frac{c^{-1}|\mathbf{e} \times \bar{\mathbf{e}}|}{|\tilde{\mathbf{e}} \times \bar{\mathbf{e}}|} = \frac{1}{c}.$$

This gives part a). For part b), see the remarks in the proof of Proposition 2 below.

2.2 Approximate Normals

Normal MTs need a rule for creating approximate normals. In our case, base points will be the midpoints of edges $\mathbf{e}$ (see Section 2.3 below), and the property we need in the proofs is the existence of a constant C_2 such that

$$\min|\pm\hat{\mathbf{n}}_{\mathbf{e}} - \hat{\mathbf{n}}_P| \le C_2|Q-P|^\alpha, \qquad \hat{\mathbf{n}}_P := (\sqrt{1+|\nabla f(P)|^2})^{-1}\mathbf{n}_P,$$

uniformly for all edges $\mathbf{e} = \mathbf{e}_{PQ}$. In other words, either $\hat{\mathbf{n}}_{\mathbf{e}}$ or $-\hat{\mathbf{n}}_{\mathbf{e}}$ should be very close to the unit normal $\hat{\mathbf{n}}_P$ to the surface Σ at the nearby point $\mathbf{v}_P$. Obviously, P can be replaced by any other point at distance $\le C|Q-P|$ from P. That this condition can be realized is guaranteed by the following

Proposition 2. *Under the conditions of Proposition 1, assume that $h(\mathbf{v}) \le \bar{h}_1$. Then the unit normal to any of the two triangular faces attached to $\mathbf{e}$ satisfies the above inequality with a constant C_2 depending on c, C_0, C_1, only. More precisely, if $\mathbf{e} = \mathbf{v}_Q - \mathbf{v}_P$ and $\tilde{\mathbf{e}} = \mathbf{v}_{\tilde{Q}} - \mathbf{v}_P$ denote edges attached to P and belonging to one of these triangular faces then we have*

$$\min|\pm\mathbf{n}_{\mathbf{e}\times\tilde{\mathbf{e}}} - \hat{\mathbf{n}}_P| \le 4C_1C_0^\alpha c^{-(1+\alpha)}|Q-P|^\alpha.$$

Here, $\mathbf{n}_{\mathbf{u}} = |\mathbf{u}|^{-1}\mathbf{u}$ denotes the unit vector in direction $\mathbf{u} \ne \mathbf{0}$.

Proof. The notation used throughout this proof and at later occasions is given in Fig. 1. Elementary computations show that

$$\begin{aligned}\mathbf{e}\times\tilde{\mathbf{e}} &= (Q-P, f(Q)-f(P)) \times (\tilde{Q}-P, f(\tilde{Q})-f(P))\\ &= (Q-P, \nabla f(P)(Q-P)+\epsilon) \times (\tilde{Q}-P, \nabla f(P)(\tilde{Q}-P)+\tilde{\epsilon})\\ &= 2A_{PQ\tilde{Q}}(-\nabla f(P),1) + \mathbf{r} = 2A_{PQ\tilde{Q}}\mathbf{n}_P + \mathbf{r},\end{aligned}$$

where $A_{PQ\tilde{Q}} = \pm\frac{1}{2}|(Q-P,0)\times(\tilde{Q}-P,0)|$ is the signed area of the triangle $PQ\tilde{Q}$ in $\mathbb{R}^2$, and the remainder term $\mathbf{r}$ can be estimated as

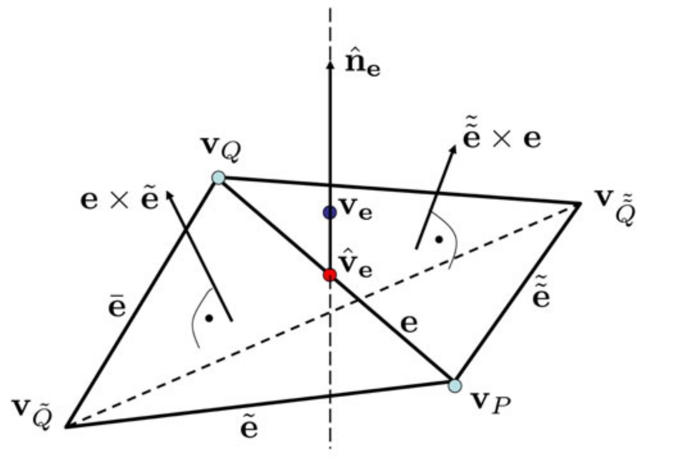
$$\begin{aligned}|\mathbf{r}|^2 &\le 2(\epsilon^2|\tilde{Q}-P|^2 + \tilde{\epsilon}^2|Q-P|^2)\\ &\le 2C_1^2|\tilde{Q}-P|^2\,|Q-P|^2(|\tilde{Q}-P|^{2\alpha} + |Q-P|^{2\alpha})\\ &\le 2C_1^2|\mathbf{e}|^2|\tilde{\mathbf{e}}|^2(|\mathbf{e}|^{2\alpha} + |\tilde{\mathbf{e}}|^{2\alpha})\\ &\le 4C_1^2|\mathbf{e}|^2|\tilde{\mathbf{e}}|^2\max(|\mathbf{e}|^{2\alpha}, |\tilde{\mathbf{e}}|^{2\alpha}).\end{aligned}$$

Since by (1) and part a) of Proposition 1

$$\max(|\mathbf{e}|,|\tilde{\mathbf{e}}|) \le C_0\max(|Q-P|,|\tilde{Q}-P|) \le C_0c^{-1}|Q-P|,$$

we arrive at

$$|\mathbf{r}| \le 2C_1|\mathbf{e}||\tilde{\mathbf{e}}|(\max(|\mathbf{e}|,|\tilde{\mathbf{e}}|))^\alpha \le 2C_1C_0^\alpha c^{-\alpha}|\mathbf{e}||\tilde{\mathbf{e}}||Q-P|^\alpha. \tag{2}$$


$\hat{\mathbf{n}}_{\mathbf{e}}$
$\tilde{\tilde{\mathbf{e}}} \times \mathbf{e}$
$\mathbf{v}_Q$
$\mathbf{e} \times \tilde{\mathbf{e}}$
$\mathbf{v}_{\mathbf{e}}$
$\mathbf{v}_{\tilde{\tilde{Q}}}$
$\hat{\mathbf{v}}_{\mathbf{e}}$
$\bar{\mathbf{e}}$
$\tilde{\tilde{\mathbf{e}}}$
$\mathbf{e}$
$\mathbf{v}_{\tilde{Q}}$
$\mathbf{v}_P$
$\tilde{\mathbf{e}}$

$$\hat{\mathbf{v}}_{\mathbf{e}} = \tfrac{1}{2}(\mathbf{v}_Q + \mathbf{v}_P), \quad \hat{\mathbf{n}}_{\mathbf{e}} = \mathbf{n}_{\mathbf{e}\times\mathbf{d}} = \mathbf{n}_{\mathbf{e}\times\tilde{\mathbf{e}}+\tilde{\tilde{\mathbf{e}}}\times\mathbf{e}}$$

Fig. 1. Notation used throughout the paper. Shown is the flap neighborhood of an edge $\mathbf{e} = \mathbf{v}_Q - \mathbf{v}_P$ in the spatial mesh, with the left triangle associated with the triangle $PQ\tilde{Q}$ in $\mathcal{T}$, the base point $\hat{\mathbf{v}}_{\mathbf{e}}$, the approximate normal $\hat{\mathbf{n}}_{\mathbf{e}}$, and the newly inserted intersection point $\mathbf{v}_{\mathbf{e}}$ of the normal line with Σ (not shown is Σ itself). The scalar detail associated with this edge is given by $d_{\mathbf{e}} = \hat{\mathbf{n}}_{\mathbf{e}}(\mathbf{v}_{\mathbf{e}} - \hat{\mathbf{v}}_{\mathbf{e}})$.

Note that this inequality yields part b) of Proposition 1 with some $\bar{c} \geq c/(2C_0)$ since for small enough $h(\mathbf{v})$

$$\sin(\angle QP\tilde{Q}) = \frac{|2A_{PQ\tilde{Q}}|}{|Q-P||\tilde{Q}-P|} \geq \frac{|\mathbf{e}\times\tilde{\mathbf{e}}| - |\mathbf{r}|}{|\mathbf{n}_P||Q-P||\tilde{Q}-P|}$$
$$\geq \frac{|\mathbf{e}||\tilde{\mathbf{e}}|(c - 2C_1 h(\mathbf{v})^{\alpha})}{C_0|Q-P||\tilde{Q}-P|} \geq \frac{c}{2C_0}.$$

The argument for the opposite direction of the statement in Proposition 1 b) is analogous, it will not be needed in this paper.

We apply next the elementary fact that

$$|\mathbf{n}_{\mathbf{u}} - \mathbf{n}_{\tilde{\mathbf{u}}}| \leq \frac{|\mathbf{u} - \tilde{\mathbf{u}}|}{\min(|\mathbf{u}|, |\tilde{\mathbf{u}}|)}$$

holds for any two non-zero directions $\mathbf{u}, \tilde{\mathbf{u}}$ to the following two particular vectors:

$$\mathbf{u} := \frac{\mathbf{e}\times\tilde{\mathbf{e}}}{2A_{PQ\tilde{Q}}}, \qquad \tilde{\mathbf{u}} := \mathbf{n}_P = (-\nabla f(P), 1).$$

Since $2A_{PQ\tilde{Q}}(\mathbf{u}-\tilde{\mathbf{u}}) = \mathbf{r}$, $2A_{PQ\tilde{Q}}\mathbf{n}_P = \mathbf{e}\times\tilde{\mathbf{e}}-\mathbf{r}$, and $\min(|\mathbf{u}|, |\tilde{\mathbf{u}}| = \min(|\mathbf{n}_P|, |\mathbf{e}\times\tilde{\mathbf{e}}|/|2A_{PQ\tilde{Q}}|$ by the notation introduced above, with the assumption $|\mathbf{e}\times\tilde{\mathbf{e}}| \geq c|\mathbf{e}||\tilde{\mathbf{e}}|$ and (2) this yields

$$\min|\pm\mathbf{n}_{\mathbf{e}\times\tilde{\mathbf{e}}} - \hat{\mathbf{n}}_P| \leq |\mathbf{n}_\mathbf{u} - \mathbf{n}_{\tilde{\mathbf{u}}}| \leq \frac{|\mathbf{r}|}{\min(|2A_{PQ\tilde{Q}}\mathbf{n}_P|, |\mathbf{e}\times\tilde{\mathbf{e}}|)}$$
$$\leq \frac{|\mathbf{r}|}{|\mathbf{e}\times\tilde{\mathbf{e}}| - |\mathbf{r}|} \leq \frac{2C_1C_0^\alpha c^{-\alpha}|\mathbf{e}||\tilde{\mathbf{e}}||Q-P|^\alpha}{|\mathbf{e}||\tilde{\mathbf{e}}|(c - 2C_1h(\mathbf{v})^\alpha)}.$$

Thus, for small enough $h(\mathbf{v})$, we arrive at the statement of Proposition 2.

Note that the slightly more symmetric with respect to $\mathbf{e}$ choice of an approximate normal $\hat{\mathbf{n}}_\mathbf{e} := \mathbf{n}_{\mathbf{e}\times\mathbf{d}}$, where $\mathbf{d} := \mathbf{v}_{\tilde{Q}} - \mathbf{v}_{\tilde{\tilde{Q}}}$ is the other diagonal of the quadrilateral composed of the two triangular faces attached to $\mathbf{e}$, will satisfy the same distance bound

$$\min|\pm\mathbf{n}_{\mathbf{e}\times\mathbf{d}} - \hat{\mathbf{n}}_P| \leq C_2|Q-P|^\alpha, \qquad C_2 = 4C_1C_0^\alpha c^{-(1+\alpha)}. \tag{3}$$

Indeed, since Proposition 2 applied to $\tilde{\tilde{\mathbf{e}}}\times\mathbf{e}$ with $\tilde{\tilde{\mathbf{e}}} = \mathbf{v}_{\tilde{\tilde{Q}}} - \mathbf{v}_P$ leads to the same estimate (and to the same sign for the area $A_{P\tilde{\tilde{Q}}Q}$), the linear combination

$$\mathbf{e}\times\tilde{\mathbf{e}} + \tilde{\tilde{\mathbf{e}}}\times\mathbf{e} = \mathbf{e}\times(\tilde{\mathbf{e}} - \tilde{\tilde{\mathbf{e}}}) = \mathbf{e}\times\mathbf{d}$$

must give it, too.

2.3 Normal MT with Edge Midpoint Prediction

For better reference, and to introduce some auxiliary notation, we first investigate a single step of the normal MT considered in this paper. Let $\mathcal{T}$ and the associated vertex set $\mathbf{v}$ on Σ be given. For each edge $\mathbf{e} = \mathbf{v}_Q - \mathbf{v}_P$, we compute base point and approximate normal:

$$\hat{\mathbf{v}}_\mathbf{e} := \frac{1}{2}(\mathbf{v}_Q + \mathbf{v}_P), \qquad \hat{\mathbf{n}}_\mathbf{e} := \mathbf{n}_{\mathbf{e}\times\mathbf{d}}. \tag{4}$$

Then we find $\mathbf{v}_e \in \Sigma$ by intersecting the line $\hat{\mathbf{v}}_\mathbf{e} + s\hat{\mathbf{n}}_\mathbf{e}$ with Σ, and denote the corresponding parameter s by $d_\mathbf{e}$ (most of this notation was already introduced in Fig. 1). This way, we obtain a new triangular mesh with vertex set $\tilde{\mathbf{v}}$ composed of $\mathbf{v}$ and the newly created points $\mathbf{v}_\mathbf{e}$, its underlying triangulation $\tilde{\mathcal{T}}$ in $\mathbb{R}^2$, and the detail vector $\tilde{d}$ composed of the individual $d_\mathbf{e}$.

The following proposition states that for c-regular triangular meshes with small enough $h(\mathbf{v})$, the intersection step is safely executable, leads to a new triangular mesh with only slightly smaller regularity constant, and also delivers the bound $|d_\mathbf{e}| = \mathrm{O}(|\mathbf{e}|^{1+\alpha})$. It is central for carrying out the recursion argument for proving our main result on normal MT in the next section.

Proposition 3. *Under the conditions of Proposition 1, there are positive constants $\bar{h}_2 \leq \bar{h}_1$ and C_3, C_4, C_5, depending on the constants c, C_0, C_1, α, such that* *for any edge* $\mathbf{e}$ *in a triangular mesh with* $h(\mathbf{v}) \leq \bar{h}_2$*, there exists a unique intersection point* $\mathbf{v}_\mathbf{e}$ *of the line segment* $\{\hat{v}_\mathbf{e} + s\hat{\mathbf{n}}_\mathbf{e} : \ |s| \leq |Q-P|/|\mathbf{n}_P|\}$ *with* Σ*. The resulting detail coefficient* $d_\mathbf{e}$ *satisfies the estimate*

$$|d_\mathbf{e}| = |\mathbf{v}_\mathbf{e} - \hat{\mathbf{v}}| \leq C_3 |Q-P|^{1+\alpha}.$$

The new triangular mesh with vertex set $\tilde{\mathbf{v}}$ *is* $\tilde{c}$*-regular with a constant*

$$\tilde{c} \geq c(1 - C_4 h(\mathbf{v})^\alpha),$$

and has mesh-width

$$h(\tilde{\mathbf{v}}) \leq \frac{1}{2}(1 + C_5 h(\mathbf{v})^\alpha) h(\mathbf{v}).$$

Proof. Since

$$\frac{1}{2}(f(P)+f(Q)) = f(P) + \frac{1}{2}\nabla f(P)(Q-P) + \epsilon, \qquad |\epsilon| \leq \frac{1}{2} C_1 |Q-P|^{1+\alpha},$$

and by the definition of $\hat{\mathbf{n}}_\mathbf{e}$ and the calculations leading to Proposition 2

$$(1+|\nabla f(P)|^2)^{1/2}\hat{\mathbf{n}}_\mathbf{e} = (-\nabla f(P), 1) + (D, \delta), \qquad \sqrt{|D|^2+\delta^2} \leq C_0 C_2 |Q-P|^\alpha,$$

we obtain the following parametrization

$$\mathbf{v}(t) = (P + (Q-P)/2 + t(-\nabla f(P) + D), f(P) + \nabla f(P)(Q-P)/2 + \epsilon + t(1+\delta))$$

with parameter $t := s|\mathbf{n}_P| \in [-|Q-P|, |Q-P|]$ of the line segment under consideration. The condition $\mathbf{v}(t) \in \Sigma$ is thus equivalent to

$$\begin{aligned} f(P) &+ \nabla f(P)(Q-P)/2 + \epsilon + t(1+\delta) \\ &= f(P + (Q-P)/2 + t(-\nabla f(P) + D)) \\ &= f(P) + \nabla f(P)((Q-P)/2 + t(-\nabla f(P)+D)) + \tilde{\epsilon}(t), \end{aligned}$$

where

$$|\tilde{\epsilon}(t)| \leq C_1 |(Q-P)/2 + t(-\nabla f(P) + D)|^{1+\alpha}.$$

After cancelation of equal terms on both sides, we obtain a scalar fix point equation

$$t = \phi(t) := \frac{\tilde{\epsilon}(t) - \epsilon}{1 + |\nabla f(P)|^2 + \delta - \nabla f(P) D}.$$

It is easy to verify that for small enough $h(\mathbf{v})$

$$\frac{d}{dt}(t - \phi(t)) \geq 0, \qquad |\phi(t)| < |Q-P|, \qquad t \in [-|Q-P|, |Q-P|].$$

For the first inequality, observe that in the interval of interest and small enough $h(\mathbf{v})$

$$\phi'(t) = \frac{\tilde{\epsilon}'(t)}{1 + |\nabla f(P)|^2 + \delta - \nabla f(P) D} = \frac{\mathrm{O}(|Q-P|^\alpha)}{1 + |\nabla f(P)|^2 + \mathrm{O}(|Q-P|^\alpha)},$$

since the definition of the above introduced function $\tilde{\epsilon}(t)$ yields

$$\begin{aligned}\tilde{\epsilon}'(t) &= \nabla f(P+(Q-P)/2+t(-\nabla f(P)+D)) - \nabla f(P))(-\nabla f(P)+D) \\ &= \mathrm{O}(|Q-P|^{\alpha}) = \mathrm{O}(h(\mathbf{v})^{\alpha}).\end{aligned}$$

Similarly, the second inequality follows from

$$|\tilde{\epsilon}(t)| \le C_1(|(Q-P)|(\frac{1}{2}+|\nabla f(P)|+|D|))^{1+\alpha} \le C_1(2C_0^2|Q-P|)^{1+\alpha}.$$

Thus, $\psi(t) = t - \phi(t)$ is a continuous, monotonously increasing function on $[-|Q-P|, |Q-P|]$, and takes opposite signs at the endpoints of this interval, which implies that $\phi(t) = t$ possesses a unique solution $t_{\mathbf{e}} = d_{\mathbf{e}}(1+|\nabla f(P)|^2)^{1/2}$. By the already available bounds on $\tilde{\epsilon}(t)$ and ϵ we get

$$\begin{aligned}|d_{\mathbf{e}}| = |\mathbf{v_e} - \hat{\mathbf{v}}_{\mathbf{e}}| \le |t_{\mathbf{e}}| = |\phi(t_{\mathbf{e}})| &\le \frac{|\tilde{\epsilon}(t_{\mathbf{e}})| + |\epsilon|}{1+|\nabla f(P)|^2 - |\delta| - |\nabla f(P)D|} \\ &\le \frac{C_1((2C_0^2)^{1+\alpha}+\frac{1}{2})|Q-P|^{1+\alpha}}{\frac{1}{2}(1+|\nabla f(P)|^2)} \le C_3|Q-P|^{1+\alpha}, \qquad (5)\end{aligned}$$

with $C_3 := C_1(1+2(2C_0)^{1+\alpha})$, if we choose $\bar{h}_2$ small enough. Observe that C_3 only depends on C_0, C_1, α, while $\bar{h}_2$ may also depend on c (e.g., via C_2).

The above estimate (5) for $|d_{\mathbf{e}}| = |\mathbf{v_e} - \hat{\mathbf{v}}_{\mathbf{e}}|$ shows the new vertex set $\tilde{\mathbf{v}}$ will be a small perturbation of the base point set $\hat{\mathbf{v}}$ obtained by edge midpoint insertion of the original triangular mesh with vertex set $\mathbf{v}$. Any two adjacent edges $\mathbf{e}'$, $\tilde{\mathbf{e}}'$ in the new mesh given by $\tilde{\mathbf{v}}$ are naturally associated with a triangle $PQ\tilde{Q}$ in $\mathcal{T}$, and with two edges $\mathbf{e}$, $\tilde{\mathbf{e}}$ of the triangular face in $\mathbf{v}$ corresponding to it so that we can write

$$\mathbf{e}' = \pm\frac{1}{2}\mathbf{e} + \mathbf{r}, \qquad \tilde{\mathbf{e}}' = \pm\frac{1}{2}\tilde{\mathbf{e}} + \tilde{\mathbf{r}}, \qquad |\mathbf{r}|, |\tilde{\mathbf{r}}| \le 2C_3\Delta^{1+\alpha}, \qquad (6)$$

where $\Delta = \max(|Q-P|, |\tilde{Q}-P|, |Q-\tilde{Q}|)$. Indeed, each edge $\mathbf{e}'$ is either connecting a newly inserted point $\mathbf{v_e}$ with one of the endpoints of $\mathbf{e} = \mathbf{v}_Q - \mathbf{v}_P$ (say $\mathbf{v}_P$) in which case by (4)

$$\mathbf{e}' = \mathbf{v_e} - \hat{\mathbf{v}}_{\mathbf{e}} + \frac{1}{2}(\mathbf{v}_P + \mathbf{v}_Q) - \mathbf{v}_P = \mathbf{r} + \frac{1}{2}\mathbf{e}$$

gives the result with $\mathbf{r} = \mathbf{v_e} - \hat{\mathbf{v}}_{\mathbf{e}}$ estimated via (5), or it connects two new points $\mathbf{v_e}$ and $\mathbf{v}_{\tilde{\mathbf{e}}}$ corresponding to two adjacent edges $\mathbf{e} = \mathbf{v}_Q - \mathbf{v}_P$ and $\tilde{\mathbf{e}} = \mathbf{v}_{\tilde{Q}} - \mathbf{v}_P$ in $\mathbf{v}$ associated with a triangle $\tilde{Q}PQ$ (note that this way any two adjacent edges $\mathbf{e}'$, $\tilde{\mathbf{e}}'$ in $\tilde{\mathbf{v}}$ determine such a triangle uniquely). In the latter case, by using again (4) we can write

$$\mathbf{e}' = \mathbf{v}_{\tilde{\mathbf{e}}} - \hat{\mathbf{v}}_{\tilde{\mathbf{e}}} + \frac{1}{2}(\mathbf{v}_{\tilde{Q}} - \mathbf{v}_Q) + \hat{\mathbf{v}}_{\mathbf{e}} - \mathbf{v_e},$$

i.e., the third edge $\bar{\mathbf{e}} = \mathbf{v}_{\tilde{Q}} - \mathbf{v}_Q$ corresponding to the triangle $\tilde{Q}PQ$ is now associated with $\mathbf{e}'$ in (6), and (5) delivers the estimate for the remainder term $\mathbf{r} = (\mathbf{v}_{\tilde{\mathbf{e}}} - \hat{\mathbf{v}}_{\tilde{\mathbf{e}}}) + (\hat{\mathbf{v}}_{\mathbf{e}} - \mathbf{v_e})$.

Since for any triangle

$$\Delta \leq \max(|\mathbf{e}|, |\tilde{\mathbf{e}}|, |\bar{\mathbf{e}}|) \leq \begin{cases} h(\mathbf{v}), \\ c^{-1} \min(|\mathbf{e}|, |\tilde{\mathbf{e}}|, |\bar{\mathbf{e}}|), \end{cases}$$

the representation (6) gives

$$|\mathbf{e}'| \leq \frac{|\mathbf{e}|}{2} + |\mathbf{r}| \leq \frac{|\mathbf{e}|}{2}(1 + \frac{4C_3 \Delta^{1+\alpha}}{|\mathbf{e}|}) \leq \frac{h(\mathbf{v})}{2}(1 + 4C_3 c^{-1} h(\mathbf{v})^{\alpha}),$$

which implies the claimed bound for $h(\tilde{\mathbf{v}})$ with $C_5 := 4C_3 c^{-1}$.

Similarly, for adjacent $\mathbf{e}', \tilde{\mathbf{e}}'$ by (6) one can write

$$\begin{aligned} \frac{|\mathbf{e}' \times \tilde{\mathbf{e}}'|}{|\mathbf{e}'||\tilde{\mathbf{e}}'|} &\geq \frac{|\mathbf{e} \times \tilde{\mathbf{e}}| - 2(|\mathbf{e}||\tilde{\mathbf{r}}| + |\tilde{\mathbf{e}}||\mathbf{r}| + 2|\tilde{\mathbf{r}}||\mathbf{r}|)}{|\mathbf{e}||\tilde{\mathbf{e}}| + 2(|\mathbf{e}||\tilde{\mathbf{r}}| + |\tilde{\mathbf{e}}||\mathbf{r}| + 2|\tilde{\mathbf{r}}||\mathbf{r}|)} \\ &\geq c\frac{1 - 2c^{-1}(|\mathbf{r}|/|\mathbf{e}| + |\tilde{\mathbf{r}}|/|\tilde{\mathbf{e}}| + 2(|\mathbf{r}||\tilde{\mathbf{r}}|)/(|\mathbf{e}||\tilde{\mathbf{e}}|))}{1 + 2(|\mathbf{r}|/|\mathbf{e}| + |\tilde{\mathbf{r}}|/|\tilde{\mathbf{e}}| + 2(|\mathbf{r}||\tilde{\mathbf{r}}|)/(|\mathbf{e}||\tilde{\mathbf{e}}|))}. \end{aligned}$$

Now substitute

$$\frac{|\mathbf{r}|}{|\mathbf{e}|} \leq 2C_3 \frac{|\Delta|^{1+\alpha}}{|\mathbf{e}|} \leq 2C_3 c^{-1} h(\mathbf{v})^{\alpha},$$

and the similar estimate for $|\tilde{\mathbf{r}}|/|\tilde{\mathbf{e}}|$ to arrive at the bound for $\tilde{c}$. This concludes the proof of Proposition 3. □

3 Main Result

We are now in a position to prove that the normal MT introduced in the previous section is well-posed for all $j \geq 1$, possesses the expected detail decay, and converges as $j \to \infty$, provided that the initial c-regular triangular mesh with vertex set $\mathbf{v}^0$ mesh has small enough mesh-width (how small depends on the constants c, C_0, C_1, and α). Convergence is understood as follows. Denote by $\tilde{\mathcal{T}}^j$ the triangulations in $\mathbb{R}^2$ obtained after j steps of uniform dyadic refinement from $\mathcal{T}^0 = \tilde{\mathcal{T}}^0$ (note that these triangulations are topologically equivalent and close to but generally different from the triangulations $\mathcal{T}^j$ associated with $\mathbf{v}^j$ if $j \geq 1$). Then we can define piecewise linear continuous vector functions $\mathbf{f}^j : \mathbb{R}^2 \to \mathbb{R}^3$ by piecewise linear interpolation of the values from $\mathbf{v}^j$ at the vertices of $\tilde{\mathcal{T}}^j$. For the given class of Σ, we call the normal MT convergent for the initial mesh $\mathbf{v}^0 \subset \Sigma$ if $\mathbf{f}^j$ converges uniformly to a continuous limit function $\mathbf{f} : \mathbb{R}^2 \to \mathbb{R}^3$ which we call normal re-parametrization of Σ.

Theorem 1. *Let Σ be a $C^{1,\alpha}$ graph surface with associated constants C_0, C_1, and fix $c \in (0, \sqrt{3}/2]$. Then there exist constants C_6, C_7, and $\bar{h} > 0$ (depending on C_0, C_1, c, α only) such that for any c-regular initial triangular mesh with vertex set $\mathbf{v}^0$ on Σ satisfying $h(\mathbf{v}^0) \leq \bar{h}$, the normal MT described in Subsection 2.3 is well-posed for all $j \geq 1$, converges to a $C^{0,1}$ re-parametrization $\mathbf{f}$ of Σ, and possesses the detail decay estimate*

$$\|d^j\|_\infty \leq C_6 2^{-(1+\alpha)j} h(\mathbf{v}^0)^{1+\alpha}, \qquad h(\mathbf{v}^j) \leq C_7 2^{-j} h(\mathbf{v}^0), \qquad j \geq 0.$$

Proof. Concerning the well-posedness of normal MT, we are almost there. The trick is to use the results from Section 2 with $\bar{c} < c$ (say $\bar{c} = c/2$), and apply Proposition 3 recursively. If c_j is the regularity constant after j successful steps of normal MT, and if $c_j \geq \bar{c}$, we can continue. Since lower bounds for c_j can be derived from Proposition 3, with a proper choice for $\bar{h}$ we will be able to guarantee this for all $j \geq 1$. Here are the details. Let us tentatively assume that we have $c_j \geq \bar{c}$ for all $j \geq 0$, and set $h_j := h(\mathbf{v}^j)$. If $h_0 \leq \bar{h}_2$ for the constant $\bar{h}_2$ determined from $\bar{c}, C_0, C_1, \alpha$ in Proposition 3 then

$$h_1 \leq \frac{1}{2}h_0(1 + C_5 h_0^\alpha) \leq \frac{1}{2}h_0(1 + C_5\bar{h}^\alpha),$$

and choosing $\bar{h} \leq \bar{h}_2$ such that also $C_5\bar{h}^\alpha \leq \frac{1}{3}$ holds, we guarantee that $h_0 \leq \bar{h}$ yields $h_1 \leq \frac{2}{3}h_0 \leq \bar{h}$. By recursion, we thus get

$$h_j \leq (\frac{2}{3})^j h_0, \qquad j \geq 1.$$

Substituting this auxiliary result into the estimates in Proposition 3, we obtain

$$h_j \leq 2^{-j}h_0 \prod_{l=0}^{j-1}(1 + C_5(\frac{2}{3})^{l\alpha}h_0^\alpha) \leq (1 + Ch_0^\alpha)2^{-j}h_0,$$

as well as

$$c_j \geq c\prod_{l=0}^{j-1}(1 - C_4(\frac{2}{3})^{l\alpha}h_0^\alpha) \geq c(1 - C'h_0^\alpha),$$

possibly after decreasing $\bar{h}$ further. Again, the constants C, C' in these bounds depend on $C_0, C_1, \bar{c}, \alpha$ only. Altogether, we see that for $\bar{c} = c/2$ there is $\bar{h}$ (possibly smaller than mandated by earlier restrictions, to also satisfy $C'\bar{h}^\alpha \leq 1/2$) such that the normal MT starting from a coarse mesh $\mathbf{v}^0$ with $h_0 \leq \bar{h}$ is well-posed for all $j \geq 1$, and satisfies the stated decay estimates for h_j and $\|d^j\|_\infty$.

To establish convergence to a $C^{0,1}$ limit $\mathbf{f}$, it remains to observe that because of the definition of $\mathbf{f}^j$ as piecewise linear interpolant of $\mathbf{v}^j$ over the dyadic refinement $\tilde{\mathcal{T}}^j$ we have

$$\|\mathbf{f}^j - \mathbf{f}^{j-1}\|_C = \sup_{\mathbf{e}} |\mathbf{v}_\mathbf{e} - \hat{\mathbf{v}}_\mathbf{e}| \leq C_3 h(\mathbf{v}^{j-1})^{1+\alpha} \leq C_3(C_7 2^{-j}h_0)^{1+\alpha},$$

where the supremum is with respect to all edges $\mathbf{e} = \mathbf{v}_Q - \mathbf{v}_P$ associated with $\mathbf{v}^{j-1}$, and notation and estimates from the proof of Proposition 3 have been recycled. This gives uniform convergence of $\mathbf{f}^j$ to $\mathbf{f}$, together with the estimate

$$\|\mathbf{f}^j - \mathbf{f}\|_C \leq C_8(2^{-j}h_0)^{1+\alpha}, \qquad j \geq 0.$$

That $\mathbf{f}$ is in $C^{0,1}$ is automatic from the fact that $\mathbf{f}^j$ interpolates the graph surface Σ given by a $C^{1,\alpha}$ function f, and that mesh regularity is already guaranteed. Obviously, better than $C^{0,1}$ regularity cannot be expected for the re-parametrization $\mathbf{f}$. We leave the detailed argument to the reader.

4 Discussion and Extensions

The above result is only a first step in the investigation of normal MTs for surfaces and other situations beyond the case of curves in $\mathbb{R}^2$. Even in the partial case considered here, some questions are left open. Do we have well-posedness and convergence also for C^1 surfaces? Is this normal MT stable (see [3,9] for the curve case)? Some applications and experimental results [14] suggest a need in investigating normal MTs for piecewise smooth (rather than globally C^1) surfaces. Finally, it is desirable to clarify the connection to research on manifold subdivision and multi-scale transforms, which builds on proximity conditions [18,5] that seem to be implicitly present in the investigation of normal MTs for smooth surfaces, too.

One of the drawbacks of normal MTs for surfaces is that due to the more complicated surface topology they may fail independently of the choice of the subdivision operator S for coarse or distorted meshes. This is the main reason for the density and regularity assumptions on $\mathbf{v}^0$ in our Theorem 1. Modifications of normal MTs to address this problem have already been discussed in [12]. In this respect, the situation for curves is better: Normal MT with linear B-spline S corresponding to edge midpoint insertion [3] resp. with the quadratic B-spline S (also called Chaikin's corner cutting subdivision) [10] converge globally, i.e., without any assumptions on $\mathbf{v}^0$, and can be used in the initial stages of the normal MT recursion to improve mesh properties before switching to the S of choice, compare [3,8]. Guaranteeing global convergence for normal MTs in the surface case remains an open problem.

To serve possible applications such as surface compression [12] or front tracking algorithms in $3D$ applications [17], it is desirable that normal MTs guarantee as fast as possible detail decay by choosing more general S. It is known from the curve case that one can expect decay estimates $\|d^j\|_\infty = \mathrm{O}(2^{-jr})$ for normal MT and $C^{k,\alpha}$ manifolds Σ ($k \geq 1$, $\alpha \in (0,1]$), if $r \leq \min(k+\alpha, s_\infty(S)+1, P_e)$, where $k+\alpha$ is the smoothness exponent of the manifold, $s_\infty(S)$ the Hölder smoothness exponent of S, and P_e its order of exact polynomial reproduction, see [3,10] (note that for some S, equality is not admissible in the above bound for r). Similar results are expected to hold for the surface case, at least away from the extraordinary vertices induced by the initial mesh. For the simple S corresponding to edge midpoint insertion considered in the previous sections, we have $s_\infty(S) = 1$, $P_e = 2$, and thus $r \leq 1+\alpha$ which is reflected in Theorem 1. Thus, improvements can only be expected for $C^{k,\alpha}$ manifolds with $k \geq 2$ and subdivision operators S with $P_e \geq 3$ and $s_\infty(S) > 1$. Examples that satisfy these conditions are the Butterfly scheme but also some interpolating schemes for $\sqrt{3}$ subdivision [6,13,11]. Loop subdivision, which was, together with the Butterfly scheme, practically tested in [12], does not provide better detail decay since for it $P_e = 2$ ($P_e \geq 3$ implies the existence of negative coefficients in the averaging rules defining S). The theoretical study of normal MTs for $C^{k,\alpha}$ graph surfaces Σ, more general S with $P_e \geq 2$ and $s_\infty(S) > 1$, and mesh topologies without extraordinary vertices is the subject of a forthcoming paper.

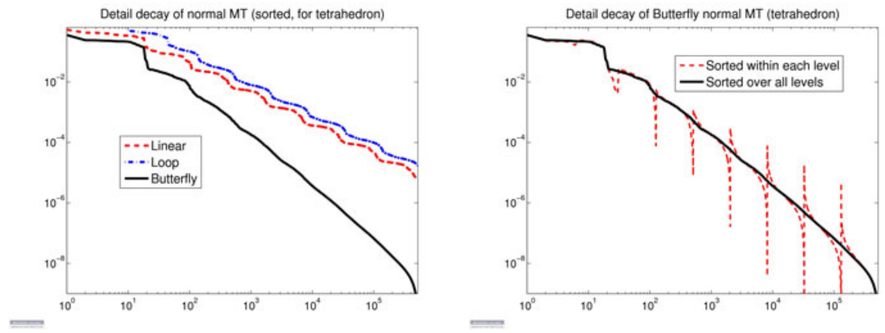


Fig. 2. Normal scheme for unit sphere with non-uniform tetrahedron at start. Left: Sorted detail plots for linear, Loop, and Butterfly normal MTs Right: Plot of $\{|d^1|, \ldots, |d^9|)$ (sorted by decreasing absolute value over all levels resp. only within each level) for the Butterfly normal MT.

Here we provide results of some preliminary numerical experiments on detail decay with normal MTs based on linear, Loop, and Butterfly subdivision operators. Our example is the unit sphere, with four points as the initial point set $\mathbf{v}^0$ located in such a way that the underlying tetrahedron is non-degenerate but non-uniform. Such an initial topology has four vertices of valence 3, and for any scheme some precaution is necessary to achieve at least C^1 smooth meshes (re-parameterizations) near these extraordinary vertices of low valence. In Fig. 2, we compare on the left the decay of the detail coefficients stored in $\{|d^1|, |d^2|, \ldots, |d^9|\}$, sorted by decreasing absolute value, for three normal MTs, namely with S defined by linear subdivision as theoretically investigated in Sections 2 and 3, Loop subdivision, and Butterfly subdivision, respectively. For the last two which were already tested in connection with surface compression in [12], extraordinary vertex treatment was implemented using the modifications originally proposed by Loop [15] and Zorin, Schröder, and Sweldens [19]. Approximate normals associated with base points corresponding to edge midpoints were computed by averaging the normals to the two attached triangular faces in the old mesh as described in Subsection 2.2 while for base point prediction close to vertices in $\mathbf{v}^{j-1}$ as needed in Loop normal MT, a similar averaging of the normals to all triangular faces attached to this vertex was performed. Evidently, Butterfly normal MT produces much faster detail decay than linear and Loop normal MT. On the right of Fig. 2, for the Butterfly normal MT we show by the dashed line the same data $\{|d^1|, |d^2|, \ldots, |d^9|\}$, but sorted only within each level. The slower decay of $\|d^j\|_\infty$ is due to the lower regularity of subdivision meshes at the four extraordinary vertices inherited from $\mathbf{v}^0$. Larger $|d_i^j|$ values remain well-localized, and no pollution effect seems to occur (this is also documented by comparing with the solid line representing the detail sequence after sorting over all levels).

Table 1 provides more numerical evidence in support of these observations. We show approximations to the decay exponents r in the expected $\mathrm{O}(2^{-jr})$

Table 1. Approximate detail decay exponents for standard normal MTs

Level	Linear		Loop		Butterfly	
j	$r_{j,\infty}$	$r_{j,2}$	$r_{j,\infty}$	$r_{j,2}$	$r_{j,\infty}$	$r_{j,2}$
2	0.5954	0.8378	0.9767	1.0404	0.5678	0.7648
3	1.9136	1.8983	1.4685	1.9634	3.0669	3.3070
4	1.8688	1.9448	1.8566	1.9735	2.4129	3.0355
5	1.9498	1.9834	1.9771	1.9857	2.1280	2.7896
6	1.9853	1.9955	2.0037	1.9959	2.0234	2.8613
7	1.9962	1.9988	1.9973	1.9990	2.0046	2.9190
8	1.9990	1.9997	1.9996	1.9997	2.0027	2.9569
9	1.9998	1.9999	1.9999	1.9999	2.0011	2.9772

estimates for two norms of the detail sequences d^j, namely maximal and mean-square detail values

$$\|d^j\|_\infty = \max_{i=1,\dots,n_j} |d_i^j|, \qquad \|d^j\|_2 = \left(n_j^{-1} \sum_{i=1}^{n_j} |d_i^j|^2 \right)^{1/2},$$

where n_j is the length of the sequence d^j. Note that n^j is smaller for interpolating than for approximating normal MTs, and is an argument in favor of using interpolating S in the construction of normal MTs. As indicator for the decay exponent characterizing the asymptotic decay of these norms, we used

$$r_{j,\alpha} := \log_2(\|d^{j-1}\|_\alpha / \|d^j\|_\alpha), \qquad j = 2, \dots, 9, \qquad \alpha = 2, \infty.$$

The values in Table 1 are in line with the predictions obtained by "extrapolating" results available from the theoretical analysis of normal MTs for curves in $\mathbb{R}^2$ to the surface case. In [3,10], it has been shown that for normal MTs of smooth curves, the decay exponent r is restricted by two factors: the limit Hölder smoothness $s_\infty(S)$ of the linear subdivision scheme S, and its order of exact polynomial reproduction P_e. More precisely, $r < \min(P_e, s_\infty(S))$, with equality possible in certain cases. Similar results are expected to hold also for the surface case, at least, away from extraordinary vertices. Since $P_e = 2$, $s_\infty(S) = 1$ for linear subdivision and $P_e = 2$, $s_\infty(S) = 3$ for Loop subdivision (this is the Hölder regularity exponent for regular triangulations, whereas the implemented modifications at extraordinary vertices guarantee $s_\infty(S) \geq 1$ in the general case), this explains the entries in the first four columns. Note that the first two columns show the sharpness of our Theorem 1 for $\alpha = 1$ since the sphere Σ is C^∞, and thus $C^{1,1}$ smooth. For the Butterfly subdivision operator we have $P_e = 4$ and $s_\infty(S) = 2$ for regular mesh topologies (as far as we know, the latter statement has not yet been proved rigorously but only supported by numerical experiments), whereas $s_\infty(S) \geq 1$ holds in the general case. Thus, we expect a decay exponent close to $\max(P_e, s_\infty(S) + 1) = 3$ in the absence of extraordinary vertices, $r \geq 2$ in general. The numerical approximations $r_{j,\infty} \approx 2$ and $r_{j,2} \approx 3$ contained in the last two columns of Table 1 perfectly match these theoretical extrapolations, and also show that the presence of extraordinary vertices affects the decay rate of the maximal detail but can be neglected on average.

Finally, in Fig. 3 and Table 2, we compare the above standard MTs with the detail decay from a new combined scheme. The rationale of this scheme is based on the following observations: The normal Loop MT produces very smooth meshes but predicts base points far off the target surface, as it reproduces only linear polynomials exactly ($P_e = 2$). Normal Butterfly meshes are not as smooth but predicted base points are relatively close to the surface since this interpolating scheme reproduces cubic polynomials exactly on uniform meshes ($P_e = 4$). Both factors (limit Hölder smoothness $s_\infty(S)$ of the meshes and order of polynomial exactness P_e) are limiting the rate of detail decay. The idea of the combined scheme is to use the normal Loop MT to find the meshes but instead of storing the relatively large details corresponding to the distance from the Loop-generated base point to the intersection point with the surface, another prediction with a scheme of high order of exact polynomial reproduction (in our case, the Butterfly scheme) should yield an improved base point along the same normal line, located at a distance asymptotically much smaller than either scheme alone could achieve. To be more precise, let S_B and S_L denote the linear Butterfly and Loop subdivision operators. Then, with $\mathbf{v}^{j-1}$ at hand, the combined scheme computes $\mathbf{v}^j$ as in the normal Loop MT by setting $\hat{\mathbf{v}}^j = S_L\mathbf{v}^{j-1}$, defining approximate normals $\hat{\mathbf{n}}^j$ as usual, and intersecting the resulting lines with Σ. What changes is the definition of details and the reconstruction formula. Denote $\tilde{\mathbf{v}}^j = S_B\mathbf{v}^{j-1}$, and define two sequences $\tilde{d}^j$, $\hat{d}^j$ entry-wise by the formulas

$$\tilde{d}^j_i = (\mathbf{v}^j_i - \tilde{\mathbf{v}}^j_i)\hat{\mathbf{n}}^j_i, \qquad \hat{d}^j_i = (\tilde{\mathbf{v}}^j_i - \hat{\mathbf{v}}^j_i)\hat{\mathbf{n}}^j_i.$$

Obviously, $d^j = \tilde{d}^j + \hat{d}^j$, and

$$\hat{\mathbf{v}}^j = \hat{\mathbf{v}}^j + (\tilde{d}^j + \hat{d}^j)\hat{\mathbf{n}}^j.$$

What is stored as normal MT data for the combined scheme is now $\tilde{d}^j$, since everything else can be recovered from $\mathbf{v}^{j-1}$, at the expense of computing the additional vector $\tilde{\mathbf{v}}^j$.

Each plot in Fig. 3 shows three curves, the sorted modified detail sequence $\{|\tilde{d}^1|, |\tilde{d}^2|, \ldots, |\tilde{d}^9|\}$ for the new combined Loop/Butterfly normal MT, the similarly defined combined Linear/Butterfly normal MT, and (for comparison) the Butterfly normal MT which was the best one among the standard schemes. In addition to result with the non-uniform tetrahedron as $\mathbf{v}^0$, we also show the test results with a symmetric double-pyramid over a regular pentagon as $\mathbf{v}^0$ (this choice yields extraordinary vertices of valence 4 and 5). Note that although the detail sequences $\tilde{d}^j$ to be stored for the new combined but non-interpolating Loop/Butterfly scheme are longer in comparison with the d^j stored for the interpolating Butterfly normal MT, the combined scheme is superior in the asymptotic range. The combination of linear normal MT with additional Butterfly-based base point prediction is not competitive due to the low limit smoothness of the associated meshes.

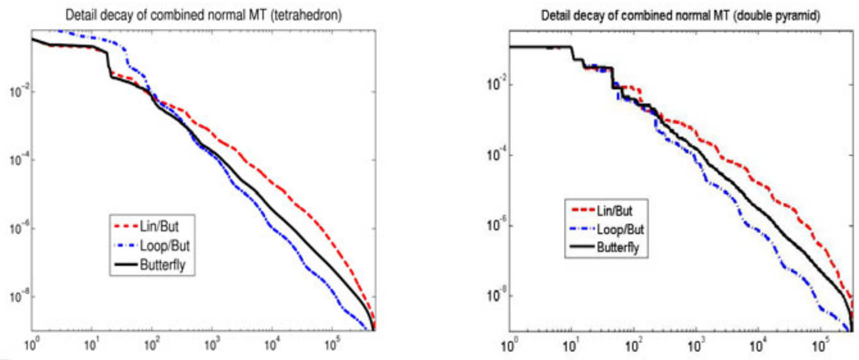


Fig. 3. Decay of detail sequences (sorted over all levels) for normal MTs (dashed line: combined Linear/Butterfly, solid line: Butterfly, dashed-dotted line: combined Loop/Butterfly schemes) for unit sphere with non-uniform tetrahedron (left) and double pyramid (right) as coarse mesh.

This is also documented by the corresponding approximations $r_{j,\alpha}$ of the decay exponents collected into Table 2 (again, the results for the standard Butterfly normal MT are included for better comparison). The observed approximations $r_{j,2}$ of decay exponents for the mean-square detail size on each level for the combined Loop/Butterfly scheme improve upon the stand-alone Loop and Butterfly normal MTs. That the asymptotic decay of maximal detail size $\|\tilde{d}^j\|_\infty$ expressed by $r_{j,\infty} \approx 4$ is as good the mean-square detail decay is somewhat surprising. For the combined Linear/Butterfly normal MT, the asymptotic behavior of these numbers is expectedly worse, and given by $r_{j,2} \approx 5/2$ and $r_{j,\infty} \approx 2$. Tests with other initial $\mathbf{v}^0$ showed similar results. The theoretical understanding of these empirical observations is still ahead.

Table 2. Approximate detail decay exponents for standard normal MTs

Level	Linear/Butt.		Loop/Butt.		Butterfly	
j	$r_{j,\infty}$	$r_{j,2}$	$r_{j,\infty}$	$r_{j,2}$	$r_{j,\infty}$	$r_{j,2}$
2	0.7359	0.8993	0.5361	0.6114	0.5678	0.7648
3	2.5279	2.8563	2.6135	3.5637	3.0669	3.3070
4	2.0309	2.3733	3.2952	3.7820	2.4129	3.0355
5	2.1528	2.4372	3.8059	3.9327	2.1280	2.7896
6	2.0436	2.4801	3.9294	3.9824	2.0234	2.8613
7	2.0104	2.4926	3.9912	3.9953	2.0046	2.9190
8	2.0026	2.4969	3.9948	3.9986	2.0027	2.9569
9	2.0006	2,4986	3.9984	3.9995	2.0011	2.9772

References

1. Baraniuk, R., Janssen, M., Lavu, S.: Multiscale approximation of piecewise smooth two-dimensional functions using normal triangulated meshes. Appl. Comput. Harm. Anal. 19, 92–130 (2005)
2. Binev, P., Dahmen, W., DeVore, R.A., Dyn, N.: Adaptive approximation of curves. In: Approximation Theory, pp. 43–57. Acad. Publ. House, Sofia (2004)
3. Daubechies, I., Runborg, O., Sweldens, W.: Normal multiresolution approximation of curves. Constr. Approx. 20, 399–462 (2005)
4. Friedel, I., Khodakovsky, A., Schröder, P.: Variational normal meshes. ACM Trans. Graph. 23, 1061–1073 (2004)
5. Grohs, P.: A general proximity analysis of nonlinear subdivision schemes. SIAM J. Math. Anal. 42, 729–750 (2010)
6. Guskov, I.: Irregular subdivision and its applications. PhD thesis, Princeton Univ., Dep. of Mathematics (1999)
7. Guskov, I., Vidimce, K., Sweldens, W., Schröder, P.: Normal meshes. In: Computer Graphics Proceedings (Siggraph 2000), pp. 95–102. ACM Press, New York (2000)
8. Harizanov, S.: Globally convergent adaptive normal multi-scale transforms. This Proceedings (submitted)
9. Harizanov, S., Oswald, P.: Stability of nonlinear subdivision and multiscale transforms. Constr. Approx. 31, 359–393 (2010)
10. Harizanov, S., Oswald, P., Shingel, T.: Normal multi-scale transforms for curves. Found. Comput. Math. 11, 617–656 (2011)
11. Jiang, J., Oswald, P.: Triangular $\sqrt{3}$-subdivision schemes: the regular case. J. Comput. Appl. Math. 156, 47–75 (2003)
12. Khodakovsky, A., Guskov, I.: Compression of normal meshes. In: Geometric Modeling for Scientific Visualization, pp. 189–207. Springer, Berlin (2003)
13. Labsik, U., Greiner, G.: Interpolatory $\sqrt{3}$-subdivision. Computer Graphics Forum 19, 131–139 (2000)
14. Lavu, S., Choi, H., Baraniuk, R.: Geometry compression of normal meshes using rate-distortion algorithms. In: Eurographics/ACM Siggraph Symposium on Geometry Processing, pp. 52–61. RWTH Aachen (2003)
15. Loop, C.: Smooth subdivision surfaces based on triangles. Master's thesis. Univ. of Utah, Dep. of Mathematics (1987)
16. Runborg, O.: Introduction to normal multiresolution analysis. In: Multiscale Methods in Science and Engineering. LNCSE, vol. 44, pp. 205–224. Springer, Heidelberg (2005)
17. Runborg, O.: Fast interface tracking via a multiresolution representation of curves and surfaces. Commun. Math. Sci. 7, 365–389 (2009)
18. Wallner, J.: Smoothness analysis of subdivision schemes by proximity. Constr. Approx. 24, 289–318 (2006)
19. Zorin, D., Schröder, P., Sweldens, W.: Interpolating subdivision for meshes of arbitrary topology. In: Computer Graphics Proceedings (SIGGRAPH 1996), pp. 189–192. ACM Press, New York (1996)

Generalized Dupin Cyclides with Rational Lines of Curvature

Martin Peternell

Institute of Discrete Mathematics and Geometry,
Vienna University of Technology,
Wiedner Hauptstrasse 8–10, 1040 Wien, Austria
http://www.dmg.tuwien.ac.at

Abstract. Dupin cyclides are algebraic surfaces of order three and four whose lines of curvature are circles. These surfaces have a variety of interesting properties and are aesthetic from a geometric and algebraic viewpoint. Besides their special property with respect to lines of curvature they appear as envelopes of one-parameter families of spheres in a twofold way. In the present article we study two families of canal surfaces with rational lines of curvature and rational principal curvatures, which contain the Dupin cyclides of order three and four as special instances in each family. The surfaces are constructed as anticaustics with respect to parallel illumination and reflection at tangent planes of curves on a cylinder of rotation.

Keywords: rational lines of curvature, canal surface, envelope of spheres, anticaustic by reflection.

1 Introduction

Dupin cyclides are among the famous surfaces studied in classical geometry and date back to the nineteenth century, see [4,6]. These surfaces are characterized by the fact that their lines of curvature are circles. These two families of circles lie in two pencils of planes and the tangent planes along a fixed circle envelope a cone of revolution. Dupin cyclides are special instances of the larger class of Darboux cyclides which denote algebraic surfaces of order four having the ideal conic $x^2 + y^2 + z^2 = 0$ as double curve. The image surfaces of quadrics in $\mathbb{R}^3$ with respect to inversion $\mathbf{x}' = \mathbf{x}/\|\mathbf{x}\|^2$ are typically Darboux cyclides. The inverse images of a cone or a cylinder of revolution or a torus is typically a Dupin cyclide.

Dupin cyclides are also quite popular in Computer Aided Geometric Design, in particular their applications for blending surfaces, see [13]. They are special instances of double Blutel conic surfaces [5], also known as supercyclides [1,14]. These surfaces carry two families of conics being contained in two pencils of planes where tangent planes along the conics form quadratic cones. The images of Dupin cyclides with respect to projective mappings are supercyclides.

A Dupin cyclide is the envelope of two one-parameter families of spheres, whose centers are contained in a pair of confocal conics. Moreover, Dupin cyclides are rational surfaces having rational offset surfaces. Thus they admit rational parameterizations with rational unit normal vectors. This property holds for all rational canal surfaces (envelopes of one-parameter families of spheres) but typically it is difficult to characterize canal surfaces with rational lines of curvature. Rational offset surfaces with rational nets of planar lines of curvature have been investigated in detail in [12]. According to this contribution there exist two classes of rational surfaces with planar lines of curvature. Their construction is based on orthogonal families of circles in the unit sphere S^2. With regard to these networks of circles in S^2, the surfaces are given by rational solutions of particular second order partial differential equations.

Contribution: We present two families of rational canal surfaces generalizing Dupin cyclides with regard to their property of having rational lines of curvature. A surface Φ of the first family possesses a rational center curve C on a rotational cylinder G and its spheres touch a cross section plane of G. Dupin cyclides of order four are obtained for ellipses $C \subset G$ as center curves, thus C is a planar section of G. The surfaces Φ of the second family generalize the Dupin cyclides of order three in a similar way. Their center curves are rational plane curves and their spheres touch a given plane which is perpendicular to the carrier plane of the center curve. All these rational canal surfaces Φ have rational offset surfaces, rational lines of curvature and rational principal curvatures. This implies that Φ has rational focal surfaces and rational Gaussian and mean curvature. Rational parameterizations of these surfaces Φ and the mentioned invariants are given explicitly.

1.1 Geometric Preliminaries

Let a surface Φ be given by the parameterization $\mathbf{f}(u,v)$, where (u,v) are coordinates in $\mathbb{R}^2$. Denoting the partial derivatives by $\mathbf{f}_u(u,v)$ and $\mathbf{f}_v(u,v)$, a normal vector of $\mathbf{f}(u,v)$ is computed by $\mathbf{n}(u,v) = \mathbf{f}_u(u,v) \times \mathbf{f}_v(u,v)$. The *first fundamental form* of $\mathbf{f}(u,v)$ is based on the scalar products of its partial derivatives,

$$\|d\mathbf{f}\|^2 = \|\mathbf{f}_u du + \mathbf{f}_v dv\|^2 = \|\mathbf{f}_u\|^2 du^2 + 2\mathbf{f}_u \cdot \mathbf{f}_v du dv + \|\mathbf{f}_v\|^2 dv^2. \qquad (1)$$

Using the abbreviations $E = \|\mathbf{f}_u\|^2$, $F = \mathbf{f}_u \cdot \mathbf{f}_v$, and $G = \|\mathbf{f}_v\|^2$, it is written as

$$\begin{aligned} \|d\mathbf{f}\|^2 = E du^2 + 2F du dv + G dv^2 &= (du, dv) \cdot \begin{pmatrix} E & F \\ F & G \end{pmatrix} \cdot (du, dv)^T \\ &= (du, dv) \cdot I(\mathbf{f}) \cdot (du, dv)^T. \end{aligned} \qquad (2)$$

The right hand side of (2) defines a local metric on the surface Φ and serves to measure lengths and areas in Φ.

Assuming $\mathbf{n}(u,v)$ to be a unit normal vector of $\mathbf{f}(u,v)$, the coefficients of the *second fundamental form* are $L = \mathbf{n} \cdot \mathbf{f}_{uu}$, $M = \mathbf{n} \cdot \mathbf{f}_{uv}$, and $N = \mathbf{n} \cdot \mathbf{f}_{vv}$. Since the identities $\mathbf{f}_u \cdot \mathbf{n} = \mathbf{f}_v \cdot \mathbf{n} = 0$ hold, L can also be expressed by $-\mathbf{f}_u \cdot \mathbf{n}_u$. Analogous expressions hold for M and N and we note that $\mathbf{n}_u$ and $\mathbf{n}_v$ are tangent vectors of Φ. The second fundamental form of $\mathbf{f}(u,v)$ reads

$$\begin{aligned} Ldu^2 + 2Mdudv + Ndv^2 &= (du,dv) \cdot \begin{pmatrix} L & M \\ M & N \end{pmatrix} \cdot (du,dv)^T \\ &= (du,dv) \cdot II(\mathbf{f}) \cdot (du,dv)^T. \end{aligned} \tag{3}$$

The principal directions of $\mathbf{f}(u,v)$ are eigenvectors of $II(\mathbf{f})$ with respect to $I(\mathbf{f})$ and the principal curvatures κ_1 and κ_2 are the respective eigenvalues. Considering a general rational surface, these functions are typically not rational.

A geometric characterization of lines of curvature on a given surface is as follows: A curve C on a surface Φ is a line of curvature if and only if the normals of Φ along C form a developable ruled surface. Lines of curvature are also characterized as surface curves having a Darboux frame which is rotation-minimizing with respect to the tangent vector of the curve, see [2].

We consider a parameterization $\mathbf{f}(u,v)$ of Φ with respect to lines of curvature which means that the u-lines as well as the v-lines are lines of curvature, thus $\mathbf{f}_u \cdot \mathbf{f}_v = 0$. Consider the developable ruled surface $\mathbf{f}(u,v^\star) + t\mathbf{n}(u,v^\star)$ formed by the normals along a u-line $\mathbf{f}(u,v^\star)$ with $v^\star = \text{const.}$, and assume $\|\mathbf{n}\|^2 = 1$. The last condition implies $\mathbf{n} \cdot \mathbf{n}_u = 0$, and we have $\mathsf{det}(\mathbf{f}_u, \mathbf{n}, \mathbf{n}_u) = 0$ and consequently $\mathbf{n}_u \cdot \mathbf{f}_v = 0$. Analogous considerations for the normals along v-lines lead to $\mathbf{n}_v \cdot \mathbf{f}_u = 0$. These properties imply that $\mathbf{n}(u,v)$ is an orthogonal net of curves in the unit sphere S^2. In case of rational offset surfaces with rational lines of curvature, $\mathbf{n}(u,v)$ is a rational orthogonal net of curves in S^2.

2 Canal Surfaces with Cylindrical Center Curve

Consider the cylinder of rotation $Z : x^2 + y^2 = 1$. A rational curve M on Z is parameterized by

$$\mathbf{m}(u) = \left(\frac{1 - f(u)^2}{1 + f(u)^2}, \frac{2f(u)}{1 + f(u)^2}, r(u) \right), \tag{4}$$

with rational functions $f(u)$ and $r(u)$. The substitution $\lambda(u) = 2 \arctan f(u)$ implies

$$\cos\lambda(u) = \frac{1 - f(u)^2}{1 + f(u)^2},\ \sin\lambda(u) = \frac{2f(u)}{1 + f(u)^2},\ \text{and}\ \dot\lambda(u) = 2\frac{\dot f(u)}{1 + f(u)^2}. \tag{5}$$

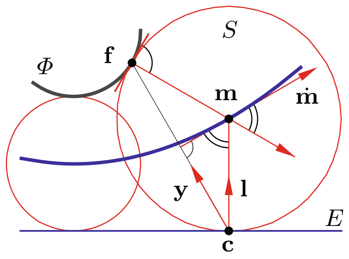
Thus, any rational curve on Z can be parameterized by

$$\mathbf{m}(u) = (\cos\lambda(u), \sin\lambda(u), r(u)). \tag{6}$$

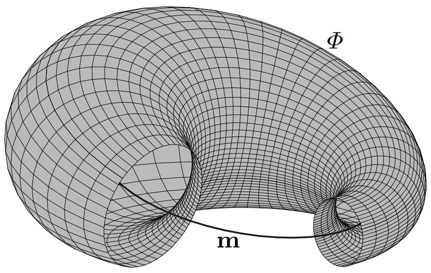
We show that the envelope Φ of the one-parameter family of spheres

$$S(u) : \|\mathbf{x} - \mathbf{m}(u)\|^2 - r(u)^2 = 0 \tag{7}$$

is a rational offset surface with rational lines of curvature. The Dupin cyclides of order four will appear as surfaces Φ for planar center curves M, see Fig. 1(b).



(a) Anticaustic mapping



(b) Dupin cyclide of degree four

Fig. 1. Anticaustic construction and Dupin cyclide

2.1 Parameterization of the Surfaces

The spheres $S(u)$ touch the plane $E : z = 0$ along the cross section curve $C : \mathbf{c}(u) = (\cos\lambda(u), \sin\lambda(u), 0)$. The characteristic circles $S(u) \cap \dot{S}(u)$ of Φ are the u-lines of the final parameterization $\mathbf{f}(u,v)$ and touch $E : z = 0$ in points of C. Φ is constructed as anticaustic by reflection with respect to light rays parallel to z. To perform this construction, we reflect points $\mathbf{c}(u)$ of C at the pencils of planes passing through the tangent lines of M, see Fig. 1(a). Here and in the following the derivatives of functions $x(u)$ are denoted by $\dot{x}$ whereas partial derivatives of bivariate functions $x(u,v)$ are denoted by x_u and x_v. Derivatives of functions $x(v)$ are denoted by x_v.

Let $\dot{\mathbf{m}}(u) = (-\dot{\lambda}\sin\lambda, \dot{\lambda}\cos\lambda, \dot{r})(u)$ be a tangent vector of $\mathbf{m}(u)$. To perform the mentioned reflection one needs to parameterize the pencil of planes through the tangent line $\mathbf{m} + t\dot{\mathbf{m}}$. A normal vector of a plane of the pencil is a linear combination of two vectors orthogonal to $\dot{\mathbf{m}}$, for instance

$$\begin{aligned} \mathbf{a}(u) &= (-\cos\lambda, -\sin\lambda, 0)(u), \text{ and} \\ \mathbf{b}(u) &= \dot{\mathbf{m}} \times \mathbf{a} = (\dot{r}\sin\lambda, -\dot{r}\cos\lambda, \dot{\lambda})(u). \end{aligned}$$

Thus these normal vectors are parameterized by $\mathbf{y}(u,v) = \gamma(u,v)\mathbf{a}(u) + \mathbf{b}(u)$, with some function $\gamma(u,v)$ to be determined. The reflection of $\mathbf{c}(u)$ at planes through the tangent lines of M is consequently given by

$$\mathbf{f}(u,v) = \mathbf{c} + 2\frac{\mathbf{y}\cdot(\mathbf{m}-\mathbf{c})}{\|\mathbf{y}\|^2}\mathbf{y}. \tag{8}$$

We still have to find a suitable function $\gamma(u,v)$ for the parameterization of $\mathbf{y}(u,v)$. The function $\mathbf{f}(u,v)$ is a parameterization with respect to lines of curvature if and only if the function $\gamma(u,v)$ satisfies

$$\dot{r}\dot{\lambda}^2 - \gamma_u\dot{\lambda} + \ddot{\lambda}\gamma = 0. \tag{9}$$

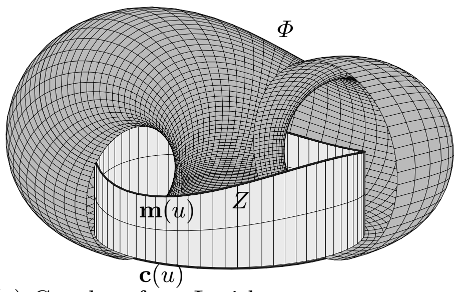
One obtains the solution $\gamma(u,v) = \dot{\lambda}(u)(r(u) + g(v))$, with some function $g(v)$. Using the abbreviation $\alpha = \dot{r}^2 + \dot{\lambda}^2(1 + (g + r)^2)$, a representation of the canal surface $\varPhi$ with respect to lines of curvature is

$$\mathbf{f}(u,v) = \frac{1}{\alpha}\begin{pmatrix} (\dot{r}^2 + \dot{\lambda}^2(1 + g^2 - r^2))\cos\lambda + 2r\dot{r}\dot{\lambda}\sin\lambda \\ (\dot{r}^2 + \dot{\lambda}^2(1 + g^2 - r^2))\sin\lambda - 2r\dot{r}\dot{\lambda}\cos\lambda \\ 2r\dot{\lambda}^2 \end{pmatrix}. \tag{10}$$

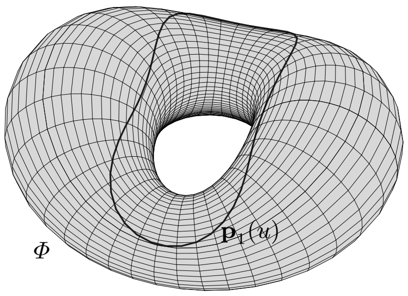
Consider rational functions $r(u)$, $f(u)$ and $g(v)$. The substitution (5) implies that (10) is a rational parameterization of $\varPhi$ with respect to its rational lines of curvature. The corresponding rational unit normal vector $\mathbf{n}(u,v)$ of $\mathbf{f}(u,v)$ satisfying $\mathbf{n}_u \cdot \mathbf{n}_v = 0$ is explicitly given by

$$\mathbf{n}(u,v) = \frac{1}{\alpha}\begin{pmatrix} 2\dot{\lambda}(\dot{\lambda}(r+g)\cos\lambda - \dot{r}\sin\lambda) \\ 2\dot{\lambda}(\dot{\lambda}(r+g)\sin\lambda + \dot{r}\cos\lambda) \\ (\dot{r}^2 + \dot{\lambda}^2(-1 + (r+g)^2) \end{pmatrix}. \tag{11}$$

Corollary 1. *Let Z be a rotational cylinder and let E be a plane perpendicular to the generating lines of Z. Consider a rational curve $M \subset Z$ and the family of spheres $S(u)$ centered at M and touching E. Then the canal surface $\varPhi$ enveloped by the spheres $S(u)$ is a rational offset surface with rational lines of curvature, explicitly represented by (10).*



(a) Canal surface $\varPhi$ with center curve $\mathbf{m}$ on a rotational cylinder Z



(b) Parabolic curve $\mathbf{p}_1$ on $\varPhi$

Fig. 2. Canal surfaces with cylindrical center curve and parabolic line

Remark: The function $g(v)$ in $\mathbf{y}(u,v) = \dot{\lambda}(u)(r(u) + g(v))\mathbf{a}(u) + \mathbf{b}(u)$ is responsible for the parameterization of the v-lines of $\mathbf{f}(u,v)$. We might set $g(v) = v$. By this simple choice the parameterization $\mathbf{f}(u,v)$ misses the cross section $\mathbf{c}(u) = (\cos u, \sin u, 0)$. This can be corrected by replacing $g(v)$ by the quotient $g(v)/h(v)$. We omitted this here to keep the formulas as simple as possible, but we will return to this idea when computing the parabolic lines of $\varPhi$ in equation (16).

2.2 Fundamental Forms and Curvatures

Since $\mathbf{f}(u,v)$ is a parameterization with respect to lines of curvature, the first and second fundamental form are both represented by diagonal matrices. With the abbreviations $\alpha = \dot{r}^2+\dot{\lambda}^2(1+(r+g)^2)$ and $\beta = \dot{\lambda}^3(1+g^2-r^2)+\dot{\lambda}\dot{r}^2+2r(\dot{r}\ddot{\lambda}-\ddot{r}\dot{\lambda})$, these matrices are

$$I(\mathbf{f}) = \frac{1}{\alpha^2}\begin{pmatrix} \beta^2 & 0 \\ 0 & 4\dot{\lambda}^4 r^2 g_v^2 \end{pmatrix}, \text{ and}$$
$$II(\mathbf{f}) = \frac{1}{\alpha^2}\begin{pmatrix} 2(\dot{r}\ddot{\lambda}-\ddot{r}\dot{\lambda}-\dot{\lambda}^3(r+g))\beta & 0 \\ 0 & 4\dot{\lambda}^4 r g_v^2 \end{pmatrix}. \tag{12}$$

The characteristic circles $S \cap \dot{S}$ of Φ are the v-lines of $\mathbf{f}(u,v)$ and are thus contained in the planes $\dot{S} : -x\dot{\lambda}\sin\lambda + y\dot{\lambda}\cos u + z\dot{r} = 0$. These planes lie in a bundle with vertex $(0,0,0)$ and thus they envelope a rational cone. The eigenvalues of II with respect to I are the principal curvatures

$$\kappa_1 = \frac{1}{r},\ \kappa_2 = \frac{2(\dot{r}\ddot{\lambda}-\ddot{r}\dot{\lambda}-\dot{\lambda}^3(r+g))}{\beta}. \tag{13}$$

It is clear that κ_1 does not depend on v and is simply the reciprocal value of the radius $r(u)$ of the spheres $S(u)$, since one family of principal curvature centers is the center curve $\mathbf{m}(u)$ itself. In case of Dupin cyclides of order four, $\mathbf{m}(u)$ is an ellipse.

The product and the mean of the principal curvatures are known to be the *Gaussian curvature* and the *mean curvature*, and are given by

$$K = \kappa_1\kappa_2 = \frac{2(\dot{r}\ddot{\lambda}-\ddot{r}\dot{\lambda}-\dot{\lambda}^3(r+g))}{r\beta}, \tag{14}$$
$$H = \frac{\kappa_1+\kappa_2}{2} = \frac{\beta+2r(\dot{r}\ddot{\lambda}-\ddot{r}\dot{\lambda}-\dot{\lambda}^3(r+g))}{2r\beta}.$$

The set of principal curvature centers of a surface typically forms two surfaces, the focal surfaces Q_1 and Q_2. These focal surfaces are obtained by measuring the reciprocal values of κ_1 and κ_2 on the surface normals $\mathbf{f}+t\mathbf{n}$, with $\|\mathbf{n}\| = 1$. In case of a canal surface Φ, one focal surface becomes the center curve $\mathbf{m}(u)$. If both focal surfaces degenerate to curves, Φ is a Dupin cyclide, thus a canal surface in a twofold way. For canal surfaces Φ, parameterized by (10) we thus obtain the center curve $\mathbf{q}_1(u) = \mathbf{f}(u,v) - r(u)\mathbf{n}(u,v) = (\cos\lambda, \sin\lambda, r)(u)$ as first component and a typically two-dimensional surface Q_2 as second component parameterized by

$$\mathbf{q}_2(u,v) = \mathbf{f}(u,v) - \frac{1}{\kappa_2(u,v)}\mathbf{n}(u,v) \tag{15}$$
$$= \frac{1}{\dot{r}\ddot{\lambda}-\ddot{r}\dot{\lambda}-\dot{\lambda}^3(g+r)}\begin{pmatrix} (\dot{r}\ddot{\lambda}-\ddot{r}\dot{\lambda})\cos\lambda - \dot{r}\dot{\lambda}^2\sin\lambda \\ (\dot{r}\ddot{\lambda}-\ddot{r}\dot{\lambda})\cos\lambda + \dot{r}\dot{\lambda}^2\cos\lambda \\ \frac{1}{2}(\dot{\lambda}^3(-1+g^2-r^2)+\dot{\lambda}\dot{r}^2+2r(\dot{r}\ddot{\lambda}-\ddot{r}\dot{\lambda})) \end{pmatrix}.$$

A point on a surface is called *parabolic* if one of its principal curvatures vanishes. The parabolic points typically form the *parabolic curves* which separate regions with elliptic and hyperbolic surface points. Since $\kappa_1(u)$ only vanishes at the poles of $r(u)$ we investigate the parabolic lines corresponding to the zeros of $\kappa_2(u,v)$. For $g = (\dot r\ddot\lambda - \ddot r\dot\lambda - r\dot\lambda^3)/\dot\lambda^3$, and with the abbreviation $\delta = \dot\lambda^4(\dot r^2 + \dot\lambda^2) + (\ddot r\dot\lambda - \dot r\ddot\lambda)^2 - 2r\dot\lambda^3(\ddot r\dot\lambda - \dot r\ddot\lambda)$ we obtain the parameterization

$$\mathbf{p}_1(u) = \frac{1}{\dot\lambda^4(\dot r^2 + \dot\lambda^2) + (\dot r\ddot\lambda - \ddot r\dot\lambda)^2}\begin{pmatrix}\delta\cos\lambda + 2r\dot r\dot\lambda^5\sin\lambda\\ \delta\sin\lambda - 2r\dot r\dot\lambda^5\cos\lambda\\ 2r\dot\lambda^6\end{pmatrix} \qquad (16)$$

of a parabolic line on Φ, see Fig. 2(b). We obviously have lost the solution $\mathbf{p}_2(u) = \mathbf{c}(u) = (\cos u, \sin u, 0)$. This is corrected by replacing $g(v)$ by $g(v)/h(v)$ in $\kappa_2(u,v)$. Its numerator becomes $2h(h(\dot r\ddot\lambda - \ddot r\dot\lambda) - \dot\lambda^3(rh+g))$ and the second parabolic line $\mathbf{c}(u)$ is obtained for $h=0$.

Theorem 1. *Let Φ be a rational canal surface given by equation (10). Φ is a rational offset surface with rational lines of curvature and the principal curvatures as well as the focal surfaces are rational. These surfaces carry two parabolic curves, the cross section C and the curve given by equation (16). The Dupin cyclides of order four are obtained for planar center curves.*

3 Canal Surfaces with Planar Center Curve

As the last section generalizes Dupin cyclides of order four, we now deal with generalizations of Dupin cyclides of order three. These cyclides are canal surfaces in a twofold way as the previous ones, but their focal curves or center curves of the families of spheres are a pair of confocal parabolas.

Consider a rational curve M in the plane Z. Without loss of generality we assume $Z : y = 0$ and let M be parameterized by

$$\mathbf{m}(u) = (q(u), 0, r(u)), \qquad (17)$$

with rational functions $q(u)$ and $r(u)$. The construction of the family of surfaces generalizing the cubic Dupin cyclides follows similar lines as the construction presented in Section 2.

3.1 Parameterization of the Surfaces

These surfaces Φ are constructed as envelopes of a one-parameter family of spheres

$$S(u) : \|\mathbf{x} - \mathbf{m}(u)\|^2 - r(u)^2 = 0, \qquad (18)$$

with center curve M and radius function r. To construct Φ as anticaustic of reflection at M with respect to light rays parallel to z, one needs to span the normal plane of $\dot{\mathbf{m}}$ by two independent vectors

$$\begin{aligned}\mathbf{a}(u) &= (0,1,0), \text{ and}\\ \mathbf{b}(u) &= \dot{\mathbf{m}} \times \mathbf{a} = (\dot r, 0, -\dot q).\end{aligned}$$

The general normal vector of the center curve is thus found by $\mathbf{y}(u,v) = \gamma(u,v)\mathbf{a} + \mathbf{b}$, with a rational function $\gamma(u,v)$. To obtain a parameterization $\mathbf{f}(u,v)$ of Φ, one performs a reflection of $\mathbf{c}(u) = (q(u),0,0)$ at all planes passing through the tangents of M, which is realized by

$$\mathbf{f}(u,v) = \mathbf{c}(u) + 2\frac{\mathbf{y}\cdot(\mathbf{m}-\mathbf{c})}{\|\mathbf{y}\|^2}\mathbf{y}. \tag{19}$$

This parameterization is a representation of Φ with respect to lines of curvature if and only if $\gamma(u,v) = g(v)\dot{q}(u)$. The explicit parameterization reads

$$\mathbf{f}(u,v) = \frac{1}{\dot{r}^2+\dot{q}^2(1+g^2)}\begin{pmatrix}(\dot{r}^2+\dot{q}^2(1+g^2))q - 2r\dot{r}\dot{q}\\ -2rg\dot{q}^2\\ 2r\dot{q}^2\end{pmatrix}, \tag{20}$$

and it is not difficult to see that these surfaces have planar lines of curvature, where one family is contained in the planes $y+gz=0$. The second family are the characteristic circles $S(u)\cap\dot{S}(u)$ and these are contained in $\dot{S}(u) : \dot{q}x+\dot{r}z = q\dot{q}$.

As in the previous section, $g(v)$ realizes the parameterization of the characteristic circles. Since we set $\mathbf{y}(u,v) = g(v)\dot{q}(u)\mathbf{a}(u)+\mathbf{b}(u)$ we miss the line $(q(u),0,0)$ in $\mathbf{f}(u,v)$. This can be corrected by replacing $g(v)$ by a quotient $g(v)/h(v)$. We omit this here to keep formulas simple but apply it later for the computation of the parabolic line.

The unit normals of $\mathbf{f}(u,v)$ are

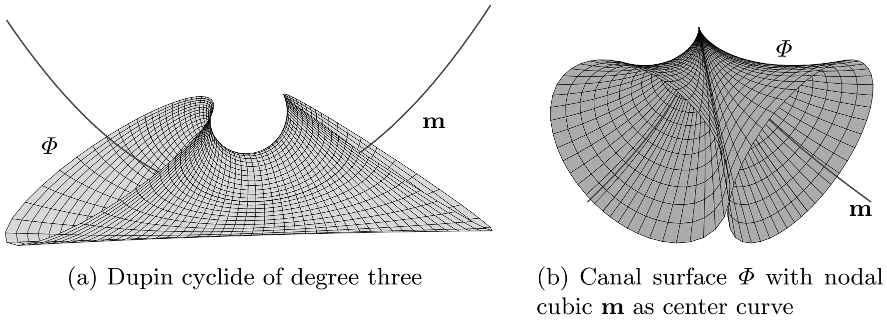
$$\mathbf{n}(u,v) = \frac{1}{\dot{r}^2+\dot{q}^2(g^2+1)}\begin{pmatrix}2\dot{q}\dot{r}\\ 2g\dot{q}^2\\ \dot{r}^2+\dot{q}^2(g^2-1)\end{pmatrix} \tag{21}$$

This parameterization of S^2 is also obtained by applying a stereographic projection $\sigma : \mathbb{R}^2 \to S^2$ with center $(0,0,1)$ to the parameterization $(\dot{r}/\dot{q}, g)$ of $\mathbb{R}^2$. The normal vectors $\mathbf{n}(u,v)$ form an orthogonal net of circles in S^2, passing through the common point $(0,0,1)$ and having orthogonal tangents there. The following result is also contained in [12] which gives a full classification of rational offset surfaces with planar rational lines of curvature.

Corollary 2. *Let Φ be a rational canal surface parameterized by (20), whose center curve M is a plane rational curve $(q(u),0,r(u))$ and whose spheres touch a line $\mathbf{c}(u) = (q(u),0,0)$. Then Φ is a rational offset surface with rational planar lines of curvature.*

3.2 Fundamental Forms and Curvatures

Analogously to Section 2, the first and second fundamental forms are both represented by diagonal matrices. Using the abbreviations $\alpha = (\dot{r}^2+\dot{q}(g^2+1))^2$ and $\beta = (-\dot{q}^3(1+g^2) - \dot{q}\dot{r}^2 + 2r(\dot{q}\ddot{r} - \ddot{q}\dot{r})$, these matrices are



(a) Dupin cyclide of degree three (b) Canal surface Φ with nodal cubic $\mathbf{m}$ as center curve

Fig. 3. Dupin cyclide of degree three and canal surface with planar center curve

$$I(\mathbf{f}) = \frac{1}{\alpha}\begin{pmatrix} \beta^2 & 0 \\ 0 & 4r^2 g_v^2 \dot{q}^4 \end{pmatrix}, \text{ and } II(\mathbf{f}) = \frac{1}{\alpha}\begin{pmatrix} 2(\dot{q}\ddot{r} - \ddot{q}\dot{r})\beta & 0 \\ 0 & 4rg_v^2\dot{q}^4 \end{pmatrix}. \tag{22}$$

The principal curvatures, the Gaussian and the mean curvature of Φ are

$$\kappa_1 = \frac{1}{r},\ \kappa_2 = \frac{2(\dot{q}\ddot{r} - \ddot{q}\dot{r})}{\beta},\ K = \frac{2(\dot{q}\ddot{r} - \ddot{q}\dot{r})}{r\beta},\ \text{and } H = \frac{\beta + 2r(\dot{q}\ddot{r} - \ddot{q}\dot{r})}{2r\beta}. \tag{23}$$

The set of focal points contains the curve $Q_1 = M$ and the two-parametric surface Q_2 which is parameterized by

$$\mathbf{q}_2(u,v) = \mathbf{f} - \frac{1}{\kappa_2}\mathbf{n} = \frac{1}{(\dot{q}\ddot{r} - \ddot{q}\dot{r})}\begin{pmatrix} q(\dot{q}\ddot{r} - \ddot{q}\dot{r}) - \dot{r}\dot{q}^2 \\ -g\dot{q}^3 \\ \frac{1}{2}(\dot{q}^3(1-g^2) - \dot{q}\dot{r}^2 + 2r(\dot{q}\ddot{r} - \ddot{q}\dot{r})) \end{pmatrix}. \tag{24}$$

Through the simple choice $g(v)$ for the parameterization of the characteristic circles we have lost the parabolic line $\mathbf{c}(u) = (q(u), 0, 0)$. By replacing $g(v)$ by $g(v)/h(v)$ in $\kappa_2(u,v)$, its numerator becomes $-2h^2(\dot{q}\ddot{r} - \ddot{q}\dot{r})$ and thus the parabolic line $\mathbf{c}(u)$ is obtained for $h = 0$.

Theorem 2. *Let Φ be a canal surface given by equation (20). Φ is a rational offset surface with rational lines of curvature and the principal curvatures as well as the focal surfaces are rational. The x-axis $\mathbf{c}(u)$ appears as parabolic line on Φ. The Dupin cyclides of order three are obtained for parabolas with z-parallel axes as center curves $\mathbf{m}(u)$ in equation (17).*

4 Conclusion

We studied particular rational canal surfaces with rational lines of curvature. These properties are invariant with respect to Möbius transformations, for instance the inversion $g : \mathbf{x}' = \mathbf{x}/\|\mathbf{x}\|^2$. Möbius transformations are conformal and preserve spheres, where planes in $\mathbb{R}^3$ are also counting as spheres (with infinite radius). Applying g to the first family of canal surfaces, one obtains canal surfaces $g(\Phi)$ with rational center curve $g(M)$ on a Dupin cyclide $g(Z)$, where the spheres $g(S)$ generating $g(\Phi)$ touch a fixed sphere $g(E)$ which is perpendicular to $g(Z)$.

Canal surfaces Φ of the second family are mapped under g to canal surfaces $g(\Phi)$ with rational spherical center curve $g(M)$ since $g(Z)$ is a sphere. The spheres $g(S)$ defining $g(Z)$ touch a fixed sphere $g(E)$ being perpendicular to $g(Z)$.

References

1. Allen, S., Dutta, D.: Supercyclides and blending. Comput. Aided Geom. Design 14, 637–651 (1997)
2. Biard, L., Farouki, R.T., Szafran, N.: Construction of rational surface patches bounded by lines of curvature. Comput. Aided Geom. Design 27, 359–371 (2010)
3. do Carmo, M.: Differential Geometry of Curves and Surfaces. Prentice-Hall, Englewood Cliffs (1976)
4. Darboux, G.: Sur une classe remarquable de courbes et de surfaces algebrique, 2nd edn., Gauthier-Villars, Paris (1896)
5. Degen, W.L.F.: Nets with Plane Silhouettes. In: The Mathematics of Surfaces V, Design and Application of Curves and Surfaces, pp. 117–133. Oxford Univ. Press, Oxford (1994)
6. Dupin, C.: Applications de Geometrie et de Mechanique. Bachelier, Paris (1822)
7. Krasauskas, R.: Branching blend of natural quadrics based on surfaces with rational offsets. Comput. Aided Geom. Design 25, 332–341 (2008)
8. Krasauskas, R., Mäurer, C.: Studying cyclides with Laguerre geometry. Comput. Aided Geom. Design 17, 101–126 (1999)
9. Peternell, M., Pottmann, H.: A Laguerre geometric approach to rational offsets. Comput. Aided Geom. Design 15, 223–249 (1998)
10. Pottmann, H., Peternell, M.: Applications of Laguerre geometry in CAGD. Comput. Aided Geom. Design 15, 165–186 (1998)
11. Pottmann, H.: Rational curves and surfaces with rational offsets. Comput. Aided Geom. Design 12, 175–192 (1995)
12. Pottmann, H., Wagner, M.: Principal Surfaces. In: The Mathematics of Surfaces VII, pp. 337–362. Information Geometers Ltd. (1998)
13. Pratt, M.J.: Cyclides in computer aided geometric design. Comput. Aided Geom. Design 7, 221–242 (1990)
14. Pratt, M.J.: Quartic supercyclides I: Basic theory. Comput. Aided Geom. Design 14, 671–692 (1997)
15. Pratt, M.J.: Cyclides in computer aided geometric design II. Comput. Aided Geom. Design 12, 131–152 (1995)
16. Srinivas, Y.L., Kumar, V., Dutta, D.: Surface design using cyclide patches. Computer-Aided Design 28, 263–276 (1996)

Spline Volume Fairing

Kjell Fredrik Pettersen and Vibeke Skytt

SINTEF ICT,
Oslo, Norway
Kjell.Fredrik.Pettersen@sintef.no,
Vibeke.Skytt@sintef.no
http://www.sintef.no

Abstract. The use of trivariate NURBS in isogeometric analysis has put the quality of parametrization of NURBS volumes on the agenda. Sometimes a NURBS volume needs a better parametrization to meet requirements regarding smoothness, approximation or periodicity. In this paper we generalize various smoothing methods that already exist for bivariate parametric spline surfaces to trivariate parametric spline volumes. We will also address how rational and polynomial spline volumes create different challenges and solutions in the algorithms.

Keywords: isogeometric analysis, spline volumes, trivariate, smoothing, approximation.

1 Introduction

Trivariate spline volumes have gained a particular interest the recent years, as they are central in isogeometric analysis, a new computational approach that integrates finite element analysis and spline objects. Isogeometric analysis was introduced by Hughes et al. [4] in order to unify the geometrical representation and analysis of new designs.

Suppose we have a spline volume, created as a Coons patch from its boundary surfaces. This approach can create unwanted effects in the volume if the shape and parametrization of these surfaces are not well adapted to this purpose. The interior control points can get a rather nasty distribution. It could even happen that the interior intersects the boundary of the volume, or that the volume has self-intersections in the interior. Also other ways to create spline volumes can result in bad volume parametrizations. If we have a volume that is closed or almost closed in a parameter direction, we may want to improve the continuity over the seam. We may also want to improve the coefficient distribution of a volume without changing it drastically, or we may want to enforce a certain parameterization on points inside a volume. The latter can occur if we want particular features in the volume to align along a constant parameter surface.

In order to find the best solution to a weighted combination of some of the above requirements, we can use spline volume fairing techniques. We construct a functional combining least squares approximations, smoothing terms and possibly some other terms depending on the purpose of the operation. The spline spaces and denominator in the rational volume function are kept fixed. Also some control points may be kept fixed, for instance at the boundaries. The remaining control points are determined by minimizing the functional. This is achieved when the gradient of the expression is zero, thus we need to solve a linear equation system. The equation system is sparse.

Fairing techniques have been used extensively on corresponding problems with spline surfaces. Several authors have published methods for surface design and surface editing based on minimizing a functional. See for instance [1,2] and [5] for methods with integration in parameter directions only, or [8] for integration in arbitrary directions in parameter space. The work in this paper is a trivariate version of [8] as we look at all directions. In addition, we lift the method to apply also to rational B-spline volumes.

In the isogeometric analysis context, volume smoothing corresponds to a mesh reparameterization. Related problems have been studied extensively within the mesh generation field. Also some CAGD related examples of volume approximation and smoothing can be found. In [3], a trilinear spline volume is defined to have an isosurface approximating a given point cloud. A shape optimization method for optimal parametrization of the computational domain for an isogeometric analysis problem with known exact solution is presented in [9]. Variational volumetric mesh grid generation with maximized smoothness is described in [6], and [7] describes a method to generate volumetric B-splines based on triangle meshes in the interior and at the boundary.

Section 2 describes the difference between NURBS and polynomial spline volumes, and presents the minimizing approach. Then we look at some specific uses of the fairing technique. Section 3 looks at point approximation, while Section 4 deals with volume smoothing where the purpose is to get the volume as smooth as possible in all parameter domain directions. We also look at how NURBS and polynomial splines give different computational orders and exactness. Some other applications, like periodicity and approximation to a default volume, are discussed in Section 5.

2 The Fairing Expression

Consider a trivariate tensor product spline volume, represented as

$$V(u_1, u_2, u_3) = \sum_{\mathbf{i}\in I} V_{\mathbf{i}}(u_1, u_2, u_3)\mathbf{c}_{\mathbf{i}}, \tag{1}$$

where the $\mathbf{c}_{\mathbf{i}} = (c^1_{\mathbf{i}}, \ldots, c^n_{\mathbf{i}}) \in \mathbf{R}^n$ are the control points of V, and $V_{\mathbf{i}}$ are the spline basis functions. We use the index set I to denote the set of triple indices

$$I = \{\mathbf{i} = (i_1, i_2, i_3) \ : \ 1 \leq i_k \leq n_k\}, \tag{2}$$

where n_1, n_2 and n_3 are the dimensions of the spline spaces in the three parameter directions. For NURBS, the rational basis functions are given as

$$V_{i_1 i_2 i_3}(u_1, u_2, u_3) = \frac{B_{i_1,d_1,\mathbf{t}_1}(u_1)B_{i_2,d_2,\mathbf{t}_2}(u_2)B_{i_3,d_3,\mathbf{t}_3}(u_3)}{\sum_{j_1,j_2,j_3} h_{j_1 j_2 j_3} B_{j_1,d_1,\mathbf{t}_1}(u_1)B_{j_2,d_2,\mathbf{t}_2}(u_2)B_{j_3,d_3,\mathbf{t}_3}(u_3)}, \tag{3}$$

where the $B_{i_k,d_k,\mathbf{t}_k}$ are the B-spline bases of the k^{th} parameter direction spline space, defined by the degree d_k and the knot vector $\mathbf{t}_k$. The weights $h_\mathbf{i}$ are all positive, ensuring the denominator function to be positive in the parameter domain. A special case occurs if all $h_\mathbf{i} = 1$, then the denominator function is 1, giving the polynomial spline volume expression

$$V(u_1, u_2, u_3) = \sum_{i_1}\sum_{i_2}\sum_{i_3} B_{i_1,d_1,\mathbf{t}_1}(u_1)B_{i_2,d_2,\mathbf{t}_2}(u_2)B_{i_3,d_3,\mathbf{t}_3}(u_3)\mathbf{c}_{ijk} \; . \tag{4}$$

In the volume fairing process, we try to find the $\mathbf{c}_\mathbf{i}, \mathbf{i} \in I_0$ for some subset $I_0 \subset I$ that minimize a weighted expression of a least squares approximation and a smoothing term. The spline spaces, denominator weights, and remaining control points $\mathbf{c}_\mathbf{j}, \mathbf{j} \in I \backslash I_0$ will be kept fixed. The square distance expression to minimize will be of the form

$$E = \sum_{\mathbf{i},\mathbf{j}\in I_0} \alpha_{\mathbf{ij}}\mathbf{c}_\mathbf{i}\mathbf{c}_\mathbf{j} + 2\sum_{\mathbf{i}\in I_0}\mathbf{p}_\mathbf{i}\mathbf{c}_\mathbf{i} + \delta = \sum_{d=1}^{n}\left(\sum_{\mathbf{i},\mathbf{j}\in I_0}\alpha_{\mathbf{ij}}c_\mathbf{i}^d c_\mathbf{j}^d + 2\sum_{\mathbf{i}\in I_0} p_\mathbf{i}^d c_\mathbf{i}^d\right) + \delta \tag{5}$$

with $\alpha_{\mathbf{ij}} = \alpha_{\mathbf{ji}}$ for all $\mathbf{i}, \mathbf{j}$. The values of $\mathbf{c}_\mathbf{i}$ that minimize E are the zeros of the gradient of E, so we need to solve the linear system $\partial E/\partial c_\mathbf{i}^d = 0$ for all $1 \le d \le n$ and $\mathbf{i} \in I_0$:

$$\mathbf{C} = A^{-1}\mathbf{P} \tag{6}$$

where $\mathbf{C}$ is the column matrix of all $\mathbf{c}_\mathbf{i}$, $\mathbf{P}$ is the column matrix of all $-\mathbf{p}_\mathbf{i}$ and A is the square symmetric matrix of all $\alpha_{\mathbf{ij}}$ for all $\mathbf{i}, \mathbf{j} \in I_0$.

In our fairing process, we will typically have several contributions to the minimizing expression:

$$E = \sum_k \omega_k E_k \tag{7}$$

for weights $\omega_k > 0$. The matrices in (6) split to

$$A = \sum_k \omega_k A_k \; , \; \mathbf{P} = \sum_k \omega_k \mathbf{P}_k \; . \tag{8}$$

The matrices A_k and A will be sparse as $[A_k]_{\mathbf{ij}} = 0$ when the supports of $V_\mathbf{i}$ and $V_\mathbf{j}$ do not overlap (with some exceptions in the case of periodicity). In the next sections, we will look for the contributions A_k and $\mathbf{P}_k$ for different terms in the fairing expression.

3 Point Approximations

One typical use of fairing techniques is to find an approximation to a set of points. We are given a finite collection

$$\{\phi_r, \mathbf{u}_r, \mathbf{q}_r\}_{r=1}^{N} \tag{9}$$

of weights ϕ_r, parameter values $\mathbf{u}_r = (u_{1,r}, u_{2,r}, u_{3,r})$ and points $\mathbf{q}_r$ in the geometry space of the volume. The task is to minimize the square distance between the points and the volume evaluation for the parameters, where we might weight the approximation importance at each point. The contribution to the minimizing expression (7) is

$$E_k = \sum_{r=1}^{N} \phi_r (V(\mathbf{u}_r) - \mathbf{q}_r)^2 \ , \tag{10}$$

which in turn gives us the matrices A_k and $\mathbf{P}_k$ in (8) by

$$[A_k]_{\mathbf{ij}} = \psi_{\mathbf{ij}} \text{ and} \tag{11}$$

$$[\mathbf{P}_k]_{\mathbf{i}} = \sum_{r=1}^{N} \phi_r V_{\mathbf{i}}(\mathbf{u}_r)\mathbf{q}_r - \sum_{\mathbf{j} \in I \setminus I_0} \psi_{\mathbf{ij}} \mathbf{c}_{\mathbf{j}} \tag{12}$$

where

$$\psi_{\mathbf{ij}} = \sum_{r=1}^{N} \phi_r V_{\mathbf{i}}(\mathbf{u}_r) V_{\mathbf{j}}(\mathbf{u}_r) \tag{13}$$

for all $\mathbf{i}, \mathbf{j} \in I$.

4 Smoothing

The most important use of volume fairing is to smoothen a volume. This is done by minimizing some derivative orders in all directions and parameter points. We will normally only be interested in derivations up to third order. A minimization of the first order derivatives tries to lower the size of the volume, and is not very relevant when the geometry space is $\mathbf{R}^3$, and the volume boundary control points are fixed. The most useful smoothing approach is related to second order derivatives. It can be compared to keeping the curvature as stable as possible for images of lines and planes in the parameter space, though it is not exactly the same; see Fig. 1. In the same way, smoothing the third order derivatives can be compared to stabilizing the change of curvature.

We want to develop the formulas by looking at derivatives in all directions and for all points, and then integrate. For a given parameter point $\mathbf{u}$ and a direction $\mathbf{s} \in S^2$ on the unit sphere, we define the k^{th} order derivative of V at $\mathbf{u}$ in direction $\mathbf{s}$ by

$$\frac{\partial^k V}{(\partial \mathbf{s})^k}(\mathbf{u}) = \frac{d^k}{(dt)^k} V(\mathbf{u} + t\mathbf{s})|_{t=0} \ ; \tag{14}$$

see Fig. 2. For $\mathbf{u} = (u_1, u_2, u_3)$ and $\mathbf{s} = (s_1, s_2, s_3)$ this becomes

$$\frac{\partial^k V}{(\partial \mathbf{s})^k}(\mathbf{u}) = \sum_{\substack{k_1,k_2,k_3 \geq 0 \\ k_1+k_2+k_3=k}} \binom{k}{k_1, k_2, k_3} s_1^{k_1} s_2^{k_2} s_3^{k_3} \frac{\partial^k V}{(\partial u_1)^{k_1} (\partial u_2)^{k_2} (\partial u_3)^{k_3}}(\mathbf{u}) \ . \tag{15}$$

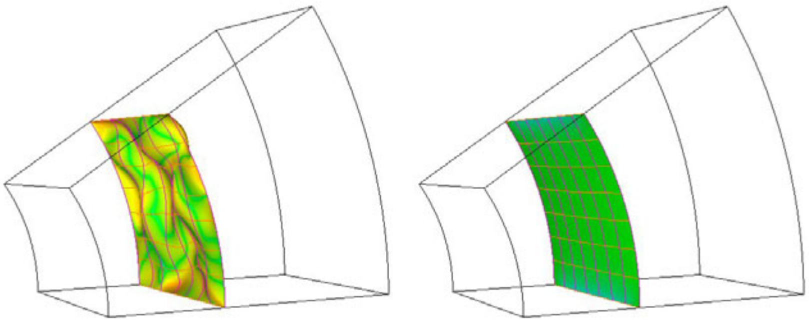


Fig. 1. To the left, a volume with an isosurface with isocurves and colors visualizing the surface curvatures before smoothing. To the right, the volume and surface after smoothing.

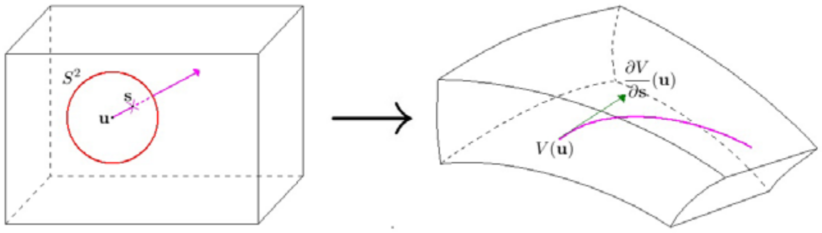


Fig. 2. The map V sends the line through $\mathbf{u}$ in direction $\mathbf{s}$ into the geometry space where its derivative at $V(\mathbf{u})$ is given as $\partial V(\mathbf{u})/\partial \mathbf{s}$

To get the term E_k in the minimization functorial (7) for the k^{th} order derivative, we square and integrate

$$E_k = \frac{1}{4\pi} \iiint_{\mathbf{u}} \iint_{\mathbf{s}} \left(\frac{\partial^k V}{(\partial \mathbf{s})^k}(\mathbf{u}) \right)^2 d\mathbf{s}\, d\mathbf{u} \tag{16}$$

over a uniform distribution of the points $\mathbf{u}$ in the parameter space and $\mathbf{s}$ on the unit sphere. The factor $1/4\pi$ is the division by the unit sphere surface area to average the sphere integration and to make the formulas simpler.

To get the contributions to the linear system, we need to simplify the expressions. For a triple index $\mathbf{m} = (m_1, m_2, m_3)$ of nonnegative integers we define

$$S^{\mathbf{m}} = \frac{1}{4\pi} \iint_{(s_1,s_2,s_3)\in S^2} s_1^{m_1} s_2^{m_2} s_3^{m_3} d(s_1, s_2, s_3) \tag{17}$$

and for triple indices $\mathbf{l}, \mathbf{m}$ such that $l_1 + l_2 + l_3 = m_1 + m_2 + m_3 = k$ and for $\mathbf{i}, \mathbf{j} \in I$ we define

$$V_{\mathbf{i},\mathbf{j}}^{\mathbf{l},\mathbf{m}} = \int_{u_1}\int_{u_2}\int_{u_3} \frac{\partial^k V_{\mathbf{i}}(u_1,u_2,u_3)}{(\partial u_1)^{l_1}(\partial u_2)^{l_2}(\partial u_3)^{l_3}} \cdot \frac{\partial^k V_{\mathbf{j}}(u_1,u_2,u_3)}{(\partial u_1)^{m_1}(\partial u_2)^{m_2}(\partial u_3)^{m_3}} du_3\, du_2\, du_1 \ . \tag{18}$$

The matrices A_k and $\mathbf{P}_k$ in (8) are then given as

$$[A_k]_{\mathbf{ij}} = \psi_{\mathbf{ij},k} \text{ and } [\mathbf{P}_k]_{\mathbf{i}} = -\sum_{\mathbf{j}\in I\setminus I_0} \psi_{\mathbf{ij},k}\mathbf{c}_{\mathbf{j}} \tag{19}$$

where

$$\psi_{\mathbf{ij},k} = \sum_{\substack{\mathbf{l},\mathbf{m} \\ l_1+l_2+l_3= \\ m_1+m_2+m_3=k}} \binom{k}{l_1,l_2,l_3}\binom{k}{m_1,m_2,m_3} S^{\mathbf{l}+\mathbf{m}} V_{\mathbf{i},\mathbf{j}}^{\mathbf{l},\mathbf{m}} \tag{20}$$

for all $\mathbf{i},\mathbf{j} \in I$. We now investigate the expressions $S^{\mathbf{l}+\mathbf{m}}$ and $V_{\mathbf{i},\mathbf{j}}^{\mathbf{l},\mathbf{m}}$ further.

There exists a well-known uniform distribution on the unit sphere by the parametrization

$$(\sqrt{1-h^2}\cos\phi, \sqrt{1-h^2}\sin\phi, h) \text{ for } h \in [-1,1] \text{ and } \phi \in [0,2\pi) \ . \tag{21}$$

We use this to get

$$S^{\mathbf{l}+\mathbf{m}} = \frac{1}{4\pi}\int_0^{2\pi} \cos^{l_1+m_1}\phi \sin^{l_2+m_2}\phi\, d\phi \int_{-1}^{1} h^{l_3+m_3}(1-h^2)^{\frac{l_1+l_2+m_1+m_2}{2}} dh \ . \tag{22}$$

This is only non-zero when l_i and m_i have the same parity for $i = 1,2,3$.

By symmetry, we can simplify further: Let $\overline{V}_{\mathbf{i},\mathbf{j}}^{\mathbf{l},\mathbf{m}}$ denote the sum of $V_{\mathbf{i},\mathbf{j}}^{\mathbf{l}',\mathbf{m}'}$ for all different pairs $(\mathbf{l}',\mathbf{m}')$ given as $\mathbf{l}' = \sigma\mathbf{l}$, $\mathbf{m}' = \sigma\mathbf{m}$ for some three-point permutation σ (example: $\overline{V}_{\mathbf{ij}}^{100,100} = V_{\mathbf{ij}}^{100,100} + V_{\mathbf{ij}}^{010,010} + V_{\mathbf{ij}}^{001,001}$). For the first, second and third order derivatives, we calculate

$$\psi_{\mathbf{ij},1} = \frac{1}{3}\overline{V}_{\mathbf{ij}}^{100,100} \tag{23}$$

$$\psi_{\mathbf{ij},2} = \frac{1}{5}\overline{V}_{\mathbf{ij}}^{200,200} + \frac{1}{15}\overline{V}_{\mathbf{ij}}^{200,020} + \frac{4}{15}\overline{V}_{\mathbf{ij}}^{110,110} \tag{24}$$

$$\psi_{\mathbf{ij},3} = \frac{1}{7}\overline{V}_{\mathbf{ij}}^{300,300} + \frac{3}{35}\overline{V}_{\mathbf{ij}}^{300,120} + \frac{9}{35}\overline{V}_{\mathbf{ij}}^{210,210} + \frac{3}{35}\overline{V}_{\mathbf{ij}}^{210,012} + \frac{12}{35}\overline{V}_{\mathbf{ij}}^{111,111} \ . \tag{25}$$

Finally we need to look at how to determine $V_{\mathbf{i},\mathbf{j}}^{\mathbf{l},\mathbf{m}}$. This is done differently for general NURBS volumes and polynomial spline volumes. The first case involves integration of piecewise rational functions, and must be done numerically, typically by Gaussian quadrature. The integration is not numerically exact, and gives an error depending on the number of sample points. For polynomial spline volumes, however, the computations can be done more efficiently and precisely by factoring the triple integral into univariate B-spline product integrals

$$V_{\mathbf{i},\mathbf{j}}^{\mathbf{l},\mathbf{m}} = \prod_{r=1}^{3}\int_{u_r} \frac{d^{l_r}}{(du_r)^{l_r}} B_{i_r,d_r,\mathbf{t}_r}(u_r) \frac{d^{m_r}}{(du_r)^{m_r}} B_{j_r,d_r,\mathbf{t}_r}(u_r)\, du_r \ . \tag{26}$$

These factors are integrals over piecewise polynomial functions. They can be precalculated, and the result is numerically exact if we use numerical integration with enough sample points.

The calculation of the linear system matrices is in general faster in the case of polynomial splines compared to NURBS. If we assume $d_i \ll n_i$, the number of pairs $(V_{\mathbf{i}}, V_{\mathbf{j}})$ of spline basis functions with common support is of order $O(\prod(n_i d_i))$. For each such pair the number of $V_{\mathbf{i},\mathbf{j}}^{\mathbf{l},\mathbf{m}}$ calculations is of order D^5 where D is the derivation depth. For polynomial splines, the B-spline product integrals are precalculated, thus the entire computation is of order

$$O(n_1 n_2 n_3 d_1 d_2 d_3 D^5) \ . \tag{27}$$

For general NURBS, we perform Gaussian quadrature with ρ_i sample points in parameter direction i inside each Bézier segment. We first precalculate $V_{\mathbf{i}}(\mathbf{u}_k)$ and all its relevant derivatives for each spline basis function and each sample point within the support of $V_{\mathbf{i}}$. The average number of common sample points for two overlapping spline basis function is of order $O(\prod(d_i \rho_i))$. The entire computation for NURBS is of order

$$O(n_1 n_2 n_3 d_1^2 d_2^2 d_3^2 \rho_1 \rho_2 \rho_3 D^5) \ . \tag{28}$$

5 Other Terms

For spline surfaces, the fairing process often starts with an input surface. We might want to modify the control points slightly to get a better parametrization without altering them too much. This can be controlled by adding an extra term that minimizes the distance between the original and derived surface. Though this is more relevant for surfaces, the method can be generalized to spline volumes. If our original volume is $V_0(\mathbf{u}) = \sum_{\mathbf{i}} \mathbf{c}_{\mathbf{i},0} V_{\mathbf{i}}(\mathbf{u})$, the minimizing term is

$$E_k = \iiint_{\mathbf{u}} \left(\sum_{\mathbf{i} \in I} V_{\mathbf{i}}(\mathbf{u})(\mathbf{c}_{\mathbf{i}} - \mathbf{c}_{\mathbf{i},0}) \right)^2 d\mathbf{u} \ . \tag{29}$$

The matrix contributions in (8) are given as

$$[A_k]_{\mathbf{ij}} = \psi_{\mathbf{ij}} \tag{30}$$

$$[\mathbf{P}_k]_{\mathbf{i}} = \sum_{\mathbf{j} \in I} \psi_{\mathbf{ij}} \mathbf{c}_{\mathbf{j},0} - \sum_{\mathbf{j} \in I \setminus I_0} \psi_{\mathbf{ij}} \mathbf{c}_{\mathbf{j}}, \tag{31}$$

where

$$\psi_{\mathbf{ij}} = \iiint_{\mathbf{u}} V_{\mathbf{i}}(\mathbf{u}) V_{\mathbf{j}}(\mathbf{u}) \, d\mathbf{u} \tag{32}$$

for all $\mathbf{i}, \mathbf{j} \in I$.

It is also possible to modify the volume to achieve periodicity over one or two seams. A C^0 gluing constraint in first parameter direction is given as

$V(u_{1,min}, u_2, u_3) = V(u_{1,max}, u_2, u_3)$ for all u_2, u_3. This is satisfied by the simple linear constraints $\mathbf{c}_{1,i_2,i_3} = \mathbf{c}_{n_1,i_2,i_3}$ for all i_2, i_3 for polynomial splines, and also for NURBS in general if the input denominator weights satisfy $h_{1,i_2,i_3} = h_{n_1,i_2,i_3}$ for all i_2, i_3.

There are several ways to get higher C^n continuity at the seam. One solution is to use periodic splines and require equality of corresponding control points. Another alternative is to use linear constraints, but this has its drawbacks. The constraints are handled by Lagrangian multipliers. This gives a saddle point problem which requires extra care if an iterative linear equation solver is used. In the case of NURBS, we also put more constraints on the weights which makes the process very inconvenient. There are methods for spline curve reparametrization to achieve a pair of equal weights at each end, see [10]. This can in some cases be generalized to surfaces and volumes, but not in general.

Instead, we choose to approximate C^n continuity at the seam by minimizing

$$\sum_{k=1}^{n} \omega_k \int_{u_2} \int_{u_3} \left(\frac{\partial^k V}{(\partial u_1)^k}(u_{1,max}, u_2, u_3) - \frac{\partial^k V}{(\partial u_1)^k}(u_{1,min}, u_2, u_3) \right)^2 \, du_2 \, du_3 \; . \tag{33}$$

As we only approximate, we are not guaranteed an exact C^n-continuity, but the method gives a more stable behavior in the equation solver than using exact constraints. We could let the weights ω_k be dominating, while we add a weaker overall smoothing constraint to ensure that nothing happens to the volume away from the gluing boundaries. This often gives C^n-continuity within a satisfactory tolerance.

As for general volume smoothing, the A_k and $\mathbf{P}_k$ matrix entries are given as inner product integrals of derivatives of basis spline volume functions. Again the computation is inaccurate for general NURBS, while the polynomial case simplifies to univariate integrals of polynomial B-spline products, being more accurate and efficient.

6 Conclusion

When we construct trivariate spline volumes, there is a risk that we end up with a bad distribution of the control points. In order to improve the result, we need a tool for reparametrization. In this paper, we have presented a way to address this issue, by using volume fairing, which is a natural extension of already existing solutions to corresponding problems for spline surfaces. A combination of least squares approximations and smoothing terms create a measure on how far away we are from an optimal solution. We minimize this measure by setting its gradient to be zero, which gives us a linear system. The solution of this system is the new set of control points.

We can use least squares approximations with respect to a finite set of points or to a given input volume V_0. In the latter case, we integrate the squared distance $(V - V_0)^2$ over the entire parameter domain of the volume.

Smoothing terms are used to minimize the derivatives of first, second or third order. We have chosen to do so in arbitratry directions in the parameter domain, by looking at $\partial^k V/(\partial \mathbf{s})^k$, the derivative of k^{th} order in any direction $\mathbf{s}$, and integrate over all directions and the entire parameter domain. The direction integrals are easy to compute when using a uniform distribution on the unit sphere. The parameter domain integrals can also be computed numerically exact for polynomial spline volumes, but in general not for NURBS. We have also described briefly how the fairing technique can be used to achieve periodicity.

We will apply these methods in our software for building models for isogeometric analysis. However, the methods presented in this paper are independent of this specific application, and can be a useful tool for many spline volume approximation and smoothing tasks.

References

1. Greiner, G., Seidel, H.-P.: Curvature Continuous Blend Surfaces. In: Proceedings of Modeling in Computer Graphics. Springer, Heidelberg (1993)
2. Hagen, H., Santarelli, P.: Variational Design of Smooth B-Spline Surfaces. Topics in Surface Modelling. SIAM, Philadelphia (1992)
3. Huang, A., Nielson, G.M.: Surface Approximation to Point Cload Data Using Volume Modeling, Data Visualization: The State of the Art, pp. 333–334. Kluwer Academic Publishers, Dordrecht (2003)
4. Hughes, T.J.R., Cottrell, J.A., Bazilevs, Y.: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. Computer Methods in Applied Mechanics and Engineering 194, 4135–4195 (2005)
5. Kallay, M.: Constrained Optimization in Surface Design. In: Proceedings of Modeling in Computer Graphics. Springer, Heidelberg (1993)
6. Knupp, P., Steinberg, S.: Fundamentals of grid generation. CRC Press, Boca Raton (1993)
7. Martin, T., Cohen, E., Kirby, R.M.: Volumetric parameterization and trivariate B-spline fitting using harmonic functions. Computer Aided Geometric Design 26, 648–664 (2009)
8. Mehlum, E., Skytt, V.: Surface Editing, Numerical Methods and Software Tools in Industrial Mathematics, pp. 381–396. Birkhäuser, Basel (1997)
9. Xu, G., Mourrain, B., Duvigneau, R., Galligo, A.: Optimal analysis-aware parameterization of computational domain in isogeometric analysis. In: Mourrain, B., Schaefer, S., Xu, G. (eds.) GMP 2010. LNCS, vol. 6130, pp. 236–254. Springer, Heidelberg (2010)
10. Lee, E.T.Y., Lucian, M.L.: Möbus reparametrizations of rational B-splines. Computer Aided Geometric Design 8, 213–215 (1991)

Neuroelectric Current Localization from Combined EEG/MEG Data

Francesca Pitolli

Dept. SBAI, University of Roma “La Sapienza”,
Via A. Scarpa 1, 00161 Roma, Italy
pitolli@dmmm.uniroma1.it
http://www.dmmm.uniroma1.it/~pitolli

Abstract. EEG/MEG devices record external signals which are generated by the neuronal electric activity of the brain. The localization of the neuronal sources requires the solution of the neuroelectromagnetic inverse problem which is highly ill-posed and ill-conditioned. We provide an iterative thresholding algorithm for recovering neuroeletric current densities within the brain through combined EEG/MEG data. We use a joint sparsity constraint to promote solutions localized in small brain area, assuming that the vector components of the current densities possess the same sparse spatial pattern. At each iteration step, the EEG/MEG forward problem is numerically solved by a Galerkin boundary element method. Some numerical experiments on the localization of current dipole sources are also given. The numerical results show that joint sparsity constraints outperform classical regularization methods based on quadratic constraints.

Keywords: Source reconstruction, Sparse representation, Thresholded iteration, Galerkin boundary element method.

1 Introduction

Functional neuroimaging aims at understanding human brain functionality through the localization of the active regions of the brain, for instance, during specific tasks or even during rest. Commonly used neuroimaging techniques, such as single-photon emission computed tomography (SPECT) and positron-emission tomography (PET), make use of invasive devices which expose the subject to x rays and radioactive tracers, respectively. Moreover, these techniques, being related to the chemical reactions that take place inside the brain, are not able to follow rapid changes occurring in neuronal activity. For this reason, in neuroscience studies there is a great effort in developing imaging techniques with higher temporal resolution.

In recent years, magnetoencephalography (MEG) has gained an important role in the field of neuroscience research since it is completely noninvasive and has a high temporal resolution. MEG aims at identifying the active area of the brain by localizing inner electric current sources through the measurements of the tiny magnetic field generated externally of the head by neuronal electric activity (see the review papers [10,20] for details). MEG is not a proper imaging technique since to recover the current distribution underlying the magnetic data an inverse problem has to be solved. It is known that the solution of the magnetic inverse problem is nonunique, since there exists silent sources which do not produce any external magnetic field [22]. Moreover, the magnetometers are just a few hundreds, while in the localization a high spatial resolution is required, and the measurements are affected by high noise. Thus, the MEG localization problem results in an highly ill-posed and ill-conditioned inverse problem and it is mandatory to use suitable regularization methods for its numerical solution [4,13,22].

Since neuronal sources are localized in small regions, the electric current flowing in the brain can be assumed to have a sparse spatial representation, i.e. it can be represented as a sum of weighted basic currents, encoded by the position and the spatial scale, in which just few terms are relevant. To promote sparse solutions, regularization methods based on sparsity constraints look promising [8,11]. In particular, since the electric current density is a vector-valued function, it makes sense to assume that all its three components have the same sparse spatial structure and to promote the same sparsity pattern on all of them. To this end, joint sparsity constraints to reconstruct multichannel signals were considered in [15,16] where an iterative thresholding algorithm to recover sparse vector-valued functions is also given. Following the same strategy, in [14] an iterative thresholding algorithm especially designed to solve the MEG inverse problem was proposed. Several numerical tests [5,25] have shown that this algorithm outperforms classical regularization methods based on quadratic constraints [4,13] in localizing current dipole sources, which are usually used to model neuroelectric currents.

In [14] just the magnetic inverse problem was considered. However, the neuronal activity generating the external magnetic field is responsible for the electric potential differences on the scalp, too. Thus, a strategy to gain some further information on neuroelectric current distributions inside the brain consists in measuring also the electric potential differences by carrying out standard (noninvasive) electroencephalografic (EEG) records during MEG measurements. Moreover, the electric and magnetic field are mutually orthogonal so that combined EEG/MEG measurements allows to detect sources which would be silent w.r.t. EEG or MEG alone. For instance, neuroelectric sources radially oriented w.r.t. the skull surface are magnetically silent but can be detected by electric measurements. On the other hand, current loops are electrically silent but produce an external magnetic field. Thus, EEG and MEG data are complementary and best results in recovering brain functionality are obtained by integrating information from both techniques [2].

In this paper we present a numerical method to solve the neuroelectromagnetic inverse problem, i.e. the reconstruction of a neuroelectric current distribution from combined EEG/MEG data, using a joint sparsity constraints as a regularization technique. The neuroelectromagnetic field can be described by the quasistatic Maxwell's equations which give rise to a boundary integral equation for the electric potential and to the Ampère-Laplace law for the magnetic field [18,19]. The integral operator equations describing the EEG/MEG forward problem will be carried out in Sec. 2. In Sec. 3 we will set the EEG/MEG inverse problem with a joint sparsity constraint and we will provide an iterative thresholding algorithm for its numerical solution that generalizes to the combined EEG/MEG inverse problem the algorithm developed in [14].

At each iteration step, we need to solve the EEG/MEG forward problem, i.e. the evaluation of the electric potential difference and the magnetic field once the neuroelectric current distribution in the brain is given. In neuroscience literature, the EEG forward problem is usually solved by a collocation boundary element method (BEM) in which the electric current density is modeled as a sum of current dipoles with given positions [2,17]. Here, we will follow a different approach: we will present a Galerkin boundary element method where we do not limit ourselves to a specific form of the current density. In this way we expect to be able to resolve not only dipole sources, but also electric current densities with different pattern, for instance, current loop that in recent studies are assumed to connect different brain regions. Details on the Galerkin BEM for the solution of the EEG/MEG forward problem will be given in Sec. 4. Finally, in Sec. 5 some numerical results on a simple test problem will be displayed and some conclusion and perspectives will be given.

2 The EEG/MEG Forward Problem

The neuroelectromagnetic field, i.e. the electromagnetic field generated by neuronal electric activity, can be describe by the Maxwell's equations. Actually, focusing on biological conductors some simplifying assumptions can be taken into account. Firstly, the permeability of the tissues in the head (scalp, skull, cerebrospinal fluid and brain) is the same as the permeability in the free space, say μ_0. Moreover, there are no electric charges in the conducting medium. Finally, due to the low frequencies of bioelectric phenomena, we may neglect the time derivatives of the electric and the magnetic fields.

Thus, the quasistatic Maxwell's equations in a tissue with electric permittivity ε_0 and magnetic permeability μ_0 read

$$\nabla \cdot \mathbf{E} = 0\,, \qquad \nabla \cdot \mathbf{B} = 0\,, \tag{1}$$

$$\nabla \times \mathbf{E} = 0\,, \qquad \nabla \times \mathbf{B} = \mu_0 \mathbf{J}\,, \tag{2}$$

where $\mathbf{E}$ and $\mathbf{B}$ are the electric and magnetic field, respectively, and $\mathbf{J}$ is the electric current density inside the tissue. As usual, we may express the irrotational electric field with a scalar potential, i.e.

$$\mathbf{E} = -\nabla V\,. \tag{3}$$

The EEG/MEG forward problem is to evaluate the neuroelectric potential V and the neuromagnetic field $\mathbf{B}$ once the electric current sources are given. It results in solving (2)-(3) once $\mathbf{J}$ is given.

The current density $\mathbf{J}$ flowing in the brain has two components. The primary current $\mathbf{J}^p$, generated by neuronal activity, flows inside or in the vicinity of the neurons. The volume current $\mathbf{J}^v(\mathbf{r}) = \sigma(\mathbf{r})\mathbf{E}(\mathbf{r})$ flows passively everywhere in the medium that is assumed to have macroscopic conductivity $\sigma(\mathbf{r})$. Thus,

$$\mathbf{J}(\mathbf{r}) = \mathbf{J}^p(\mathbf{r}) + \sigma(\mathbf{r})\mathbf{E}(\mathbf{r}) = \mathbf{J}^p(\mathbf{r}) - \sigma(\mathbf{r})\nabla V(\mathbf{r}) \,. \tag{4}$$

Following [18],[19] we model the head as a conductor consisting of homogeneous nested regions, each one having constant conductivity. Let G_i, $i = 0, \ldots, m$, be the regions and σ_i, $i = 0, \ldots, m$, be the constant conductivity inside G_i. Let us denote by S_i, $i = 0, \ldots, m$, the interfaces between G_i and G_{i+1} with $\mathbf{n}_i(\mathbf{r})$ being the unit vector perpendicular to ∂G_i in $\mathbf{r}$ and pointing at G_{i+1}. We assume the regions are nested with G_0 being the innermost region, i.e.

$$G_0 \subset G_1 \subset \cdots \subset G_m \,,$$

and the interfaces S_i, $i = 0, \ldots, m$, are not intersecting. The primary current $\mathbf{J}^p$ flows just inside G_0.

From the Maxwell's equations (see [20] for details) it follows that the electric current density (4) and the external magnetic field are related by the Ampère-Laplace law that reads

$$\mathbf{B}(\mathbf{r}) = \mathbf{B}_0(\mathbf{r}) - \frac{\mu_0}{4\pi} \sum_{i=1}^{m} \sigma_i \int_{G_i \setminus G_{i-1}} \nabla V(\mathbf{r}') \times \frac{\mathbf{r} - \mathbf{r}'}{|\mathbf{r} - \mathbf{r}'|^3} \, d\mathbf{r}' \,, \tag{5}$$

where

$$\mathbf{B}_0(\mathbf{r}) = \frac{\mu_0}{4\pi} \int_{G_0} \frac{\mathbf{J}^p(\mathbf{r}') \times (\mathbf{r} - \mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|^3} \, d\mathbf{r}' \tag{6}$$

is the magnetic field produced by $\mathbf{J}^p$ in an infinite homogeneous medium. The volume integrals in (5) can be transformed into surface integrals on the interfaces [19] obtaining

$$\mathbf{B}(\mathbf{r}) = \mathbf{B}_0(\mathbf{r}) - \frac{\mu_0}{4\pi} \sum_{i=0}^{m} (\sigma_{i+1} - \sigma_i) \int_{S_i} V(\mathbf{r}') \frac{\mathbf{r} - \mathbf{r}'}{|\mathbf{r} - \mathbf{r}'|^3} \times \mathbf{n}_i(\mathbf{r}') \, dS_i(\mathbf{r}') \,, \tag{7}$$

(in the sum we may assume $\sigma_{m+1} = 0$).

It can be shown [20] that the electric potential V satisfies a surface integral equation which involves V at the interfaces only:

$$\frac{(\sigma_{l+1} + \sigma_l)}{2} V(\mathbf{r}) = \sigma_0 V_0(\mathbf{r}) + \frac{1}{4\pi} \sum_{i=1}^{m} (\sigma_{i+1} - \sigma_i) \int_{S_i} V(\mathbf{r}') \, \frac{\mathbf{r} - \mathbf{r}'}{|\mathbf{r} - \mathbf{r}'|^3} \cdot \mathbf{n}_i(\mathbf{r}') \, dS_i(\mathbf{r}') \,, \quad \mathbf{r} \in S_l, \; l = 0, \ldots, m, \tag{8}$$

where

$$V_0(\mathbf{r}) = \frac{1}{4\pi\sigma_0} \int_{G_0} \frac{\mathbf{J}^p(\mathbf{r'}) \cdot (\mathbf{r} - \mathbf{r'})}{|\mathbf{r} - \mathbf{r'}|^3} \, d\mathbf{r'} \tag{9}$$

is the electric potential produced by $\mathbf{J}^p$ in an infinite homogeneous medium [18].

Equations (6)-(9) can be solved analytically for a few simple current distributions and conductor geometry. For instance, an analytical solution of the forward problem can be obtained when the sources are modeled as current dipoles in a homogeneous conducting sphere [26]. More realistic head models require numerical methods to solve the integral equation (8) and to evaluate (7). This will be addressed in Sec. 4.

3 The EEG/MEG Inverse Problem

In order to gain information on brain functionality we need to reconstruct the primary current density $\mathbf{J}^p$ from the measured data. This results in an inverse problem, known as neuroelectromagnetic inverse problem, which is to estimate the brain current sources underlying the measurements of the scalp electric potential and the external magnetic field [10,20].

Let $\mathbf{q}_i$, $i = 1, \ldots, M$, and $\mathbf{p}_i$, $i = 1, \ldots, N$, be the magnetometer and electrode sites, respectively. The sites $\mathbf{q}_i$, $i = 1, \ldots, M$, belong to a surface Σ with $dist(\Sigma, G_m) > 0$, and each magnetometer measures the magnetic field B_i, $i = 1, \ldots, M$, along the direction $\mathbf{e}(\mathbf{q}_i)$ (usually the normal to the magnetometer coil). The sites $\mathbf{p}_i$, $i = 1, \ldots, N$, belong to the surface G_m (the scalp), and each electrode measures the potential difference V_i, $i = 1, \ldots, N$, w.r.t. a reference electrode.

Now, let $\mathcal{B}_{\mathbf{e}}(\mathbf{r}, \mathbf{J}^p) := \mathbf{B}(\mathbf{r}) \cdot \mathbf{e}(\mathbf{r})$ and $\mathcal{V}(\mathbf{r}, \mathbf{J}^p) := V(\mathbf{r})$ be the integral operators representing the solution of the forward problem (7)-(8). We note that both $\mathcal{B}_{\mathbf{e}}(\mathbf{r}, \mathbf{J}^p)$ and $\mathcal{V}(\mathbf{r}, \mathbf{J}^p)$ are linearly related to $\mathbf{J}^p$. Thus, the neuroelectromagnetic inverse problem is to minimize the *discrepancy*

$$\Delta(\mathbf{J}^p) = \sum_{i=1}^{M} \big(B_i - \mathcal{B}_{\mathbf{e}}(\mathbf{q}_i, \mathbf{J}^p)\big)^2 + \sum_{i=1}^{N} \big(V_i - \mathcal{V}(\mathbf{p}_i, \mathbf{J}^p)\big)^2 \tag{10}$$

w.r.t. to the primary current distribution $\mathbf{J}^p$.

We recall that a current distribution inside a conductor cannot be retrieved uniquely from knowledge of the electromagnetic field outside (see, for instance, [20,22]). There are primary current distributions that are either magnetically, or electrically silent, or both, i.e. there may exist neuronal currents that do not produce any external magnetic field or electric potential differences on the head. Thus, we must add some further constraints in order to confine ourselves to find a solution among a limited class of source configurations [4,13].

The regularization methods based on quadratic constraints lead to over smoothed source estimations [8,12]. In particular, minimum ℓ_2-norm estimates, i.e. minimizers of the functional

$$\Delta(\mathbf{J}^p) + \alpha \, \|\mathbf{J}^p\|_2^2 \, , \qquad \alpha > 0 \, , \tag{11}$$

are not appropriate for the localization of epileptic foci, which are known to be confined in small brain regions. This allows us to assume that the primary current has a *sparse representation* w.r.t. a suitable basis of compactly supported functions $(\psi_\lambda)_{\lambda\in\Lambda}$, i.e.

$$\mathbf{J}^p = (J^1, J^2, J^3) \in L_2(G_0;\mathbb{R}^3)\,, \qquad J^\ell = \sum_{\lambda\in\Lambda} j^\ell_\lambda\, \psi_\lambda\,, \qquad \ell = 1,2,3\,, \tag{12}$$

where only few coefficients (j^ℓ_λ) for each component are non-vanishing [11].

We note that multiscale basis are successfully used in medical signal processing and image analysis since scale adaptivity allows to detect spikes and discontinuities appearing in biological signals or inhomogeneous structures characterizing biological tissues [1]. Thus, we may assume $(\psi_\lambda)_{\lambda\in\Lambda}$ be a multiscale basis, i.e. a wavelet basis or a frame [23].

Since all the components J^ℓ, $\ell = 1,2,3$, are related to the same neurophysiological phenomenon, it makes sense to assume the subset of non-vanishing coefficients being the same for all the three components. This is equivalent to require that $\mathbf{J}^p$ has a sparse representation w.r.t. the joint ℓ_q-norm, defined as

$$\|(\mathbf{j}_\lambda)_{\lambda\in\Lambda}\|_q := \left(\sum_{\lambda\in\Lambda}\left(\|\mathbf{j}_\lambda\|_{\mathbb{R}^3}\right)^q\right)^{1/q}, \qquad q \geq 1\,, \tag{13}$$

where $\mathbf{j}_\lambda = (j^1_\lambda, j^2_\lambda, j^3_\lambda)^T$ [15].

Let $\mathbf{j} = (\mathbf{j}_\lambda)_{\lambda\in\Lambda}$. Following [15], it can be shown that the solution of the EEG/MEG inverse problem with a joint sparsity constraint is the minimizer of the functional

$$\mathcal{J}^{(q)}_{\theta,\rho,\omega}(\mathbf{j}, v) := \Delta(\mathbf{j}) + \left(\sum_{\lambda\in\Lambda} v_\lambda\, \|\mathbf{j}_\lambda\|_q + \sum_{\lambda\in\Lambda}\omega_\lambda\, \|\mathbf{j}_\lambda\|_2^2 + \sum_{\lambda\in\Lambda}\theta_\lambda(\rho_\lambda - v_\lambda)^2\right), \tag{14}$$

restricted to $v_\lambda \geq 0$. Here, $(\theta_\lambda)_{\lambda\in\Lambda}$, $(\rho_\lambda)_{\lambda\in\Lambda}$, and $(\omega_\lambda)_{\lambda\in\Lambda}$ are some suitable positive parameter sequences. The discrepancy $\Delta(\mathbf{j})$ can be obtained by inserting in (10) the current density representation (12) so that

$$\Delta(\mathbf{j}) = \|F - T_\psi\, \mathbf{j}\|_2^2\,, \tag{15}$$

where $F = [(V_i)_{i=1,\ldots,N}, (B_i)_{i=1,\ldots,M}]^T$ denotes the measurement vector, while T_ψ denotes the matrix whose entries are the coefficients of the operators $\mathcal{V}(\mathbf{r}, \mathbf{J}^p)$ and $\mathcal{B}_\mathbf{e}(\mathbf{r}, \mathbf{J}^p)$ w.r.t. $(j^\ell_\lambda)_{(\lambda\in\Lambda)(\ell=1,2,3)}$ (see Sec. 4 for details).

The task is to minimize $\mathcal{J}_{\theta,\rho,\omega}(\mathbf{j}, v)$ jointly with respect to both the variables $\mathbf{j}$ and v. The first belongs to the space of signals (current densities) to be reconstructed, the second belongs to the space of sparsity indicator weights (see [15,16] for details).

The minimizer $(\mathbf{j}^*, v^*)$ of the functional $\mathcal{J}^{(q)}_{\theta,\rho,\omega}$ subject to the joint sparsity constraints can be approximated by the following iterative algorithm, deduced from [16] (see also [14]).

Vector Iterative Thresholding Algorithm

$$\begin{cases} \text{Let } \gamma \text{ be a suitable relaxation parameter} \\ \text{Choose an arbitrary} \quad \mathbf{j}^{(0)} \in \ell_2(\boldsymbol{\Lambda}; \mathrm{I\!R}^3) \\ \text{For} \quad 0 \le k \le K \quad \text{do} \quad \mathbf{j}^{(k+1)} = \mathcal{S}^{(q)}_{\theta,\rho,\omega}\left(\mathbf{j}^{(k)} + \gamma\, T^*_\psi (F - T_\psi\, \mathbf{j}^{(k)})\right) \end{cases} \tag{16}$$

The operator $\mathcal{S}^{(q)}_{\theta,\rho,\omega} : \ell_2(\boldsymbol{\Lambda}; \mathrm{I\!R}^3) \to \ell_2(\boldsymbol{\Lambda}; \mathrm{I\!R}^3)$ is the vector-valued thresholding operator introduced in [15] and can be efficiently evaluated by the algorithm given in [16].

The convergence of the algorithm above can be proved as in [14], nevertheless we expect a slow convergence rate, as already put in evidence in some numerical tests on the solution of a magnetic bidimensional inverse problem [25]. More efficient algorithms can be obtained by choosing an adaptive relaxation parameter. Some strategies to speed-up the convergence rate of Alg. (16) can be found in [3],[5],[9].

4 Discretization of the Forward Problem

In order to implement Alg. (16) we need to solve efficiently the integral equation for $V(\mathbf{r})$ at each iteration step. In neuroscience literature the forward problem is usually solved by a collocation BEM assuming that the primary current $\mathbf{J}^p$ can be represented by a sum of current dipoles (see, for instance, [2,17]). Here, we discretize the boundary integral equations (8) by a Galerkin BEM [21] assuming $\mathbf{J}^p$ has the sparse representation (12).

To formulate the Galerkin BEM for the EEG/MEG problem let us introduce the integral operators:

$$\begin{aligned}
(\mathcal{L}_B \mathbf{J}^p)(\mathbf{r}) &:= \frac{\mu_0}{4\pi} \int_{G_0} \frac{\mathbf{J}^p(\mathbf{r}') \times (\mathbf{r} - \mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|^3} \cdot \mathbf{e}(\mathbf{r})\, d\mathbf{r}' \\
&= -\frac{\mu_0}{4\pi} \int_{G_0} \frac{\mathbf{e}(\mathbf{r}) \times (\mathbf{r} - \mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|^3} \cdot \mathbf{J}^p(\mathbf{r}')\, d\mathbf{r}'\,, \\
(\mathcal{S}_B V)(\mathbf{r}) &:= -\frac{\mu_0}{4\pi} \sum_{i=0}^{m} (\sigma_{i+1} - \sigma_i) \int_{S_i} V(\mathbf{r}') \frac{\mathbf{r} - \mathbf{r}'}{|\mathbf{r} - \mathbf{r}'|^3} \times \mathbf{n}_i(\mathbf{r}') \cdot \boldsymbol{e}(\mathbf{r})\, dS_i(\mathbf{r}')\,, \\
(\mathcal{L}_E \mathbf{J}^p)(\mathbf{r}) &:= \frac{\sigma_0}{4\pi(\sigma_{l+1} + \sigma_l)} \int_{G_0} \frac{\mathbf{J}^p(\mathbf{r}') \cdot (\mathbf{r} - \mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|}\, d\mathbf{r}'\,, \\
(\mathcal{S}_E V)(\mathbf{r}) &:= \frac{1}{4\pi} \sum_{i=0}^{m} \frac{(\sigma_{i+1} - \sigma_i)}{(\sigma_{l+1} + \sigma_l)} \int_{S_i} V(\mathbf{r}')\, \frac{\mathbf{r} - \mathbf{r}'}{|\mathbf{r} - \mathbf{r}'|^3} \cdot \mathbf{n}_i(\mathbf{r}')\, dS_i(\mathbf{r}')\,.
\end{aligned} \tag{17}$$

From (8)-(9) it follows that the unknown function $V(\mathbf{r}) \in \mathcal{H} = L_2(\cup_{i=0}^{m} \partial G_i)$ satisfies the Fredholm integral equation of the second kind

$$\big([\frac{1}{2}I - \mathcal{S}_E]\, V\big)(\mathbf{r}) = (\mathcal{L}_E \mathbf{J}^p)(\mathbf{r})\,, \qquad \mathbf{r} \in \partial G_l\,, \quad l = 0, \ldots, m\,. \tag{18}$$

The integral operator for the component of the magnetic field along the direction $\mathbf{e}(\mathbf{r})$ can be deduced from (6)-(7),

$$B(\mathbf{r}) = \mathbf{B}(\mathbf{r}) \cdot \mathbf{e}(\mathbf{r}) = (\mathcal{L}_B \mathbf{J}^p)(\mathbf{r}) + (\mathcal{S}_B\, V)(\mathbf{r})\,, \qquad \mathbf{r} \in \mathbb{R}^3 \setminus \cup_{i=0}^{m} G_i\,. \tag{19}$$

The Galerkin BEM consists in finding an element

$$V_{\mathcal{H}_\lambda}(\mathbf{r}) = \sum_{\lambda \in \Lambda} v_\lambda\, \psi_\lambda(\mathbf{r})\,, \tag{20}$$

belonging to the space

$$\mathcal{H}_\lambda = span\big((\psi_\lambda)_{\lambda \in \Lambda}\big) \subset \mathcal{H}\,,$$

such that

$$\sum_{\lambda \in \Lambda} v_\lambda\, \langle (\frac{1}{2}I - \mathcal{S}_E)\, \psi_\lambda, \psi_\mu \rangle = \langle \mathcal{L}_E \mathbf{J}^p, \psi_\mu \rangle\,, \tag{21}$$

for all ψ_μ, $\mu \in \Lambda$. Classical results on boundary element methods (see [21]) allow us to conclude that the Galerkin equations (21) have a unique solution. Moreover, the numerical solution is quasi-optimal, i.e. the following error estimate holds

$$\|V - V_{\mathcal{H}_\lambda}\|_{\mathcal{H}} \leq c \inf_{V_{\mathcal{H}_\lambda} \in \mathcal{H}_\lambda} \|V - V_{\mathcal{H}_\lambda}\|_{\mathcal{H}}\,. \tag{22}$$

The unknown coefficient vector $Y = [v_\lambda]_{\lambda \in \Lambda}$ is the solution to the linear system

$$A_E\, Y = B_E\,, \tag{23}$$

where

$$A_E = \big[\langle (\frac{1}{2}I - \mathcal{S}_E)\, \psi_\lambda, \psi_\mu \rangle\big]_{\mu, \lambda \in \Lambda}\,, \tag{24}$$

and

$$B_E = \big[\langle \mathcal{L}_E \mathbf{J}^p, \psi_\mu \rangle\big]_{\mu \in \Lambda}\,. \tag{25}$$

For later use it is more convenient to factorize B_E as

$$B_E = L_E\, J\,, \tag{26}$$

where

$$L_E = \big[\langle \mathcal{L}_E \mathbf{e}^\ell\, \psi_\lambda, \psi_\mu \rangle\big]_{(\mu, \lambda \in \Lambda)(\ell = 1,2,3)}\,, \tag{27}$$

with $\mathbf{e}^\ell$, $\ell = 1, 2, 3$, being the unitary versors of the coordinate system, and

$$J = \big[j_\lambda^\ell\big]_{(\lambda \in \Lambda)(\ell = 1,2,3)}\,. \tag{28}$$

Once the approximation $V_{\mathcal{H}_\lambda}$ is evaluated, the magnetic field can be approximated by

$$B_{\mathcal{H}_\lambda}(\mathbf{r}) = (\mathcal{L}_B \mathbf{J}^p)(\mathbf{r}) + (\mathcal{S}_B\, V_{\mathcal{H}_\lambda})(\mathbf{r})\,. \tag{29}$$

The entries of the matrix T_ψ in Alg. (16) are the coefficients of $B_{\mathcal{H}_\lambda}(\mathbf{r})$ and $V_{\mathcal{H}_\lambda}(\mathbf{r})$ w.r.t. $\mathbf{j} = (j_\lambda)_{(\lambda\in\Lambda)(\ell=1,2,3)}$. To give the explicit expression of T_ψ, let us write $V_{\mathcal{H}_\lambda}$ and $B_{\mathcal{H}_\lambda}$ as a function of $\mathbf{j}$. From (20), (23) and (26) it follows that

$$V_{\mathcal{H}_\lambda}(\mathbf{r}) = \sum_{\lambda\in\Lambda}\sum_{\mu\in\Lambda}\sum_{\ell=1}^{3} (A_E^{-1}L_E)^\ell_{\mu\lambda}\, j^\ell_\mu\, \psi_\lambda(\mathbf{r})\,. \tag{30}$$

Now, inserting (12) and (30) in (29) we get

$$B_{\mathcal{H}_\lambda}(\mathbf{r}) = \sum_{\lambda\in\Lambda}\sum_{\ell=1}^{3} (L_B(\mathbf{r}))^\ell_\lambda\, j^\ell_\lambda + \sum_{\lambda\in\Lambda}\sum_{\mu\in\Lambda}\sum_{\ell=1}^{3} (A_E^{-1}L_E)^\ell_{\lambda\mu}\, j^\ell_\mu\, (\mathcal{S}_B\psi_\lambda)(\mathbf{r})\,, \tag{31}$$

where

$$L_B(\mathbf{r}) = \left[(\mathcal{L}_B \mathbf{e}^\ell\, \psi_\lambda)(\mathbf{r})\right]_{(\lambda\in\Lambda)(\ell=1,2,3)}\,. \tag{32}$$

Let

$$T_E = \left[\sum_{\mu\in\Lambda}(A_E^{-1}L_E)^\ell_{\mu\lambda}\,\psi_\mu(\mathbf{p}_i)\right]_{(i=1,\ldots,N)(\lambda\in\Lambda,\ell=1,2,3)} \tag{33}$$

and

$$T_B = \left[(L_B(\mathbf{q}_i))^\ell_\lambda\,\psi_\lambda(\mathbf{q}_i) + \sum_{\mu\in\Lambda}(A_E^{-1}L_E)^\ell_{\mu\lambda}\,(\mathcal{S}_B\psi_\mu)(\mathbf{q}_i)\right]_{(i=1,\ldots,M)(\lambda\in\Lambda,\ell=1,2,3)}, \tag{34}$$

by (30) and (31) we obtain

$$T_\psi = \begin{bmatrix} T_E \\ T_B \end{bmatrix} \tag{35}$$

We note that Alg. (16) can be implemented efficiently if the matrix $T^*_\psi T_\psi$ can be approximated by a sparse finite matrix. Sparse representations of the integral operators (17) can be obtained by using multiscale bases [6],[7],[24].

5 Numerical Tests

To give an idea of the behavior of Alg. (16) we consider a simple test problem, i.e. the localization of current dipole sources in a homogeneous spherical conductor G_0 with radius $R = 10cm$. In this case the surface integrals in (29) vanish and the magnetic and electric problems decouple. Nevertheless, the integration of electric and magnetic data allows us to improve the source localization accuracy.

In these tests, the synthetic electromagnetic data are generated by three current dipoles located at a depth of $0.1R$ below the surface of the sphere. One

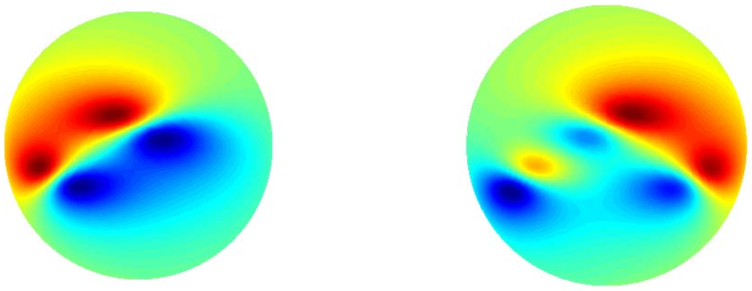


Fig. 1. Synthetic electric potential (left) and magnetic field (right)

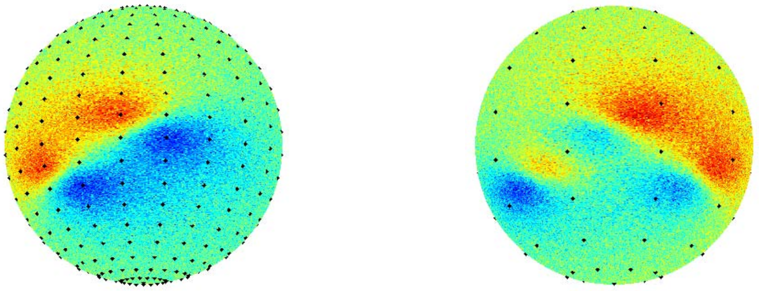


Fig. 2. Synthetic noisy electric potential (left) and magnetic field (right). The sensor sites are displayed as black points.

dipole is radially oriented, so that it does not produce any external magnetic field. The behavior of the electric potential and the radial component of the magnetic field is shown in Fig. 1. To obtain the synthetic measurements, the radial component of the magnetic field has been sampled on $M = 400$ sites, distributed on a sphere of radius $1.1R$ concentric to G_0, while the electric potential has been sampled on $N = 100$ sites, located on the sphere surface. The electric and magnetic data are scaled in order to have the same norm. Finally, a white noise with linear *snr* equal to 1 has been added. The synthetic noisy data and the sensor distribution are shown in Fig. 2.

The spherical conductor has been parametrized in a spherical coordinate system and a linear finite element space with 64^3 degrees of freedom has been used as approximation space. The inverse problem has been solved by the iterative thresholding algorithm (16) with 20 iteration steps. Finally, the acceleration strategy proposed in [3] has been used to speed-up the convergence rate.

In Fig. 3 (left) the intensity of the current density reconstructed by Algorithm (16) is displayed. For comparison, the results obtained by classical Tikhonov regularization are also shown (Fig. 3, right). A qualitative analysis of the figures show that the proposed algorithm allows us to better focus the current sources, while Tikhonov regularization produces a more blurred image with fictitious sources.

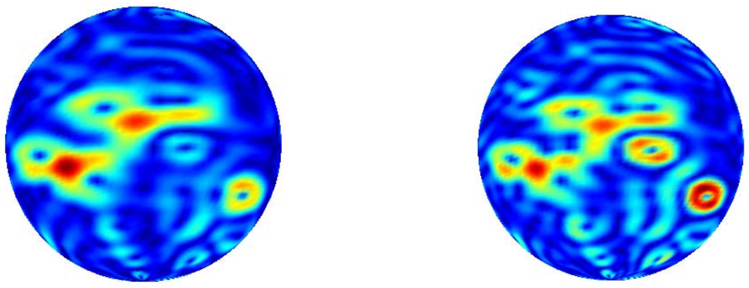


Fig. 3. The intensity of the reconstructed current density obtained by Alg. (16) (left) and by Tikhonov regolarization (right) when using the noisy data

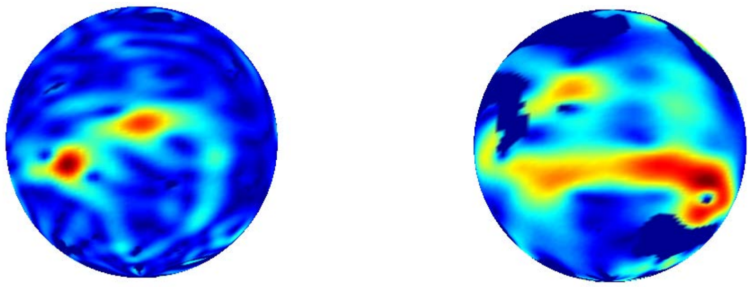


Fig. 4. The intensity of the reconstructed current density obtained by using just magnetic (left) or electric (right) noisy data in Alg. (16)

In Tab. 1 the localization error (LE) and the spatial dispersion (SD) are shown for each dipole source. Let σ be the source location and $I(\mathbf{r})$ be the intensity of the reconstructed current density. The LE is the distance between σ and the location of the maximum of $I(\mathbf{r})$ in the neighborhood of the source; the SD is evaluated as $\|d(\mathbf{r})I(\mathbf{r})\|_2/\|I(\mathbf{r})\|_2$. Smaller the values of LDE and SD, the higher accuracy and smaller spread.

It is interesting to observe that magnetic data or electric data give rise to inaccurate localization when used uncoupled (see Fig. 4).

Even if this preliminary experiment has a very simple geometry and uses a single approximation scale, the results show that both integration of electric and magnetic data and joint sparsity allow to improve localization accuracy. Next step will be to implement the algorithm on a real head geometry and to solve the EEG/MEG problem using real high-level noisy data. When we deal with real-life applications, we face some difficulties that require more sophisticated numerical techniques. First of all spherical coordinate system may present singularities, so that we need more efficient parametrization of the head. Moreover, it would be useful to dispose of multiscale bases especially designed for representing neuroelectric currents. Finally, the processing of high noise electro- and magneto-encephalographic signals require suitable estimator.

Table 1. Localization error (LE) and spatial dispersion (SD)

	Alg. (16)	Tikhonov method
LDE (first source)	2.5 mm	3.4 mm
SD (first source)	0.06 mm	0.48 mm
LDE (second source)	1.9 mm	3.5 mm
SD (second source)	0.15 mm	0.22 mm
LDE (third source)	8.0 mm	9.7 mm
SD (third source)	0.18 mm	0.31 mm

References

1. Aldoubi, A., Unser, M., Laine, A. (eds.): Wavelets in Biomedical Imaging. IEEE Trans. on Medical Imaging 22, 285–288 (2003)
2. Babiloni, F., Carducci, F., Cincotti, F., Del Gratta, C., Pizzella, V., Romani, G.L., Rossini, P.M., Tecchio, F., Babiloni, C.: Linear Inverse Source Estimate of Combined EEG and MEG Data Related to Voluntary Movements. Hum. Brain Mapp. 14, 197–209 (2001)
3. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. 2, 183–202 (2009)
4. Bertero, M., Boccacci, P.: Introduction to Inverse Problems in Imaging. Institute of Physics, Bristol (2002)
5. Bretti, G., Fornasier, M., Pitolli, F.: Electric current density imaging via an accelerated iterative algorithm with joint sparsity constraints. In: SPARS 2009, St-Malo, France (2009), http://hal.inria.fr/SPARS09/en
6. Cohen, A., Hoffmann, M., Reiss, M.: Adaptive wavelet Galerkin methods for linear inverse problems. SIAM J. Numer. Anal. 42, 1479–1501 (2004)
7. Dahmen, W., Harbrecht, H., Schneider, R.: Adaptive methods for boundary integral equations: complexity and convergence estimates. Math. Comp. 76, 1243–1274 (2007)
8. Daubechies, I., Defrise, M., De Mol, C.: An Iterative Thresholding Algorithm for Linear Inverse Problems with a Sparsity Constraint. Commun. Pure Appl. Math. 57, 1413–1457 (2004)
9. Daubechies, I., Fornasier, M., Loris, I.: Accelerated Projected Gradient Method for Linear Inverse Problems with Sparsity Constraints. J. Fourier Anal. Appl. 14, 764–792 (2008)
10. Del Gratta, C., Pizzella, V., Tecchio, F., Romani, G.L.: Magnetoencephalography - A Noninvasive Brain Imaging Method with 1ms Time Resolution. Rep. Prog. Phys. 64, 1759–1814 (2001)
11. Donoho, D.L.: Superresolution via Sparsity Constraints. SIAM J. Math. Anal. 23, 1309–1331 (1992)
12. Donoho, D.L.: De-noising by Soft-Thresholding. IEEE Trans. Inform. Theory 41, 613–627 (1995)
13. Engl, H.W., Hanke, M., Neubauer, A.: Regularization of Inverse Problems. Kluwer, Dordrecht (2000)
14. Fornasier, M., Pitolli, F.: Adaptive Iterative Thresholding Algorithms for Magnetoencephalography (MEG). J. Comput. Appl. Math. 221, 386–395 (2008)

15. Fornasier, M., Rauhut, H.: Iterative Thresholding Algorithms. Appl. Comput. Harmon. Anal. 25, 187–208 (2008)
16. Fornasier, M., Rauhut, H.: Recovery Algorithms for Vector Valued Data with Joint Sparsity Constraints. SIAM J. Numer. Anal. 46, 577–613 (2008)
17. Fuchs, M., Wagner, M., Kastner, J.: Boundary Element Method Volume Conductor Models for EEG Source Reconstruction. Clin. Neurophysiol. 112, 1400–1407 (2001)
18. Geselowitz, D.B.: On bioelectric potentials in an inhomogeneous volume conductor. Biophys. J. 7, 1–11 (1967)
19. Geselowitz, D.B.: On the magnetic field generated outside an inhomogeneous volume conductor by internal current sources. IEEE Trans. Magn. 6, 346–347 (1970)
20. Hamalainen, M.S., Hari, R., Ilmoniemi, R.J., Knuuttila, J., Lounasmaa, O.V.: Magnetoencephalography - Theory, Instrumentation and Applications to Non Invasive Studies of the Working Buman brain. Rev. Mod. Phys. 65, 413–497 (1993)
21. Hsiao, G.C., Wendland, W.L.: Boundary Element Methods: Foundation and Error Analysis. In: Stein, E., de Borst, R., Hughes, T.J.R. (eds.) Encyclopedia of Computational Mechanics, pp. 339–373. John Wiley & Sons, New York (2004)
22. Kaipio, J., Somersalo, E.: Statistical and computational inverse problems. In: Applied Mathematical Sciences, vol. 160. Springer, New York (2005)
23. Mallat, S.: A Wavelet Tour on Signal Processing. The Sparse Way, 3rd edn. Elsevier/Academic Press (2009)
24. von Petersdorff, T., Schwab, C., Schneider, R.: Multiwavelets for second-kind integral equations. SIAM J. Numer. Anal. 34, 2212–2227 (1997)
25. Pitolli, F., Bretti, G.: An Iterative Algorithm with Joint Sparsity Constraints for Magnetic Tomography. In: Dæhlen, M., Floater, M., Lyche, T., Merrien, J.-L., Mørken, K., Schumaker, L.L. (eds.) MMCS 2008. LNCS, vol. 5862, pp. 316–328. Springer, Heidelberg (2010)
26. Sarvas, J.: Basic Mathematical and Electromagnetic Concepts of the Biomagnetic Inverse Problem. Phys. Med. Biol. 32, 11–22 (1987)

Bootstrap-Based Normal Reconstruction

Ahmad Ramli and Ioannis Ivrissimtzis

School of Engineering and Computing Sciences, Durham University, UK

Abstract. We propose a bootstrap-based method for normal estimation on an unorganised point set. Experimental results show that the accuracy of the method is comparable with the accuracy of the widely used Principal Component Analysis. The main advantage of our approach is that the variance of the normals over the bootstrap samples can be used as a confidence value for the estimated normal. In a proposed application, we use the confidence values to construct a bilateral Gaussian filter for normal smoothing.

Keywords: bootstrap, normal reconstruction, bilateral filter, normal smoothing.

1 Introduction

The geometric modelling of a natural object starts with the data acquisition stage, where geometric data are obtained, usually in the form of unorganised 3D point sets. After the registration of the data, and possibly some preprocessing for outlier removal and denoising, a mathematical model of the object is reconstructed, usually in the form of a polygonal mesh or an implicit surface. At this stage, several of the state-of-the-art surface reconstruction algorithms require accurate estimations of normal vectors associated with the points of the set. The process of estimating such normals is known as *normal reconstruction*.

In this paper we propose a method for normal reconstruction based on *bootstrap*. Bootstrap is a general statistical technique widely used for model fitting and assessment. The bootstrap process first randomly subsamples the input data set, creating several subsets, called *bootstrap samples*. Then, a model is fitted to each of the bootstrap samples and finally, properties of the input data are inferred from statistical aspects of the bootstrap models. Statistics can be computed for model features or characteristics.

In our context of normal reconstruction, the initial data set is the k nearest neighborhood of a point d_i of the point set. The model fitted in each bootstrap sample is an estimated tangent plane computed with Principal Component Analysis (PCA). From each tangent plane we compute a normal vector and the average of these normals is our normal estimate at d_i. The variance of the angles of the bootstrap normals is also computed and treated as a confidence value for the normal at d_i.

Our experiments show that the bootstrap normals are comparable with the commonly used PCA normals and thus their main advantage lies in the confidence values associated with them. As an application of these confidence values, we construct a bilateral Gaussian filter for normal smoothing. In this filter, the influence of d_i on d_j depends not only on the distance between d_i and d_j, but on the normal bootstrap variance at d_i as well.

Summarising, the main contributions of the paper are:

– A bootstrap-based method for normal reconstruction which produces normals with confidence values.
– A bilateral Gaussian filter for normal smoothing utilising the confidence values of the bootstrap normals.

1.1 Related Work

As reliable normal estimations are essential for many geometry processing and rendering algorithms, the literature on normal reconstruction is extensive. The classic method proposed in [7] applies PCA on the k-neighbourhood of a point. The result of is equivalent to computing the normal of the minimum least square fitting plane for the same neighbourhood. Due to its simplicity and robustness, [7] is still widely used and considered to be part of the state-of-the-art. [13] improves on PCA by computing a weighted least square fitting plane, with weights given by sigmoidal functions. [5] uses Singular Value Decomposition to compute a normal that minimises the variance of the dot products between itself and the vectors from the considered point to its nearest neighbours. [12] applies least square fitting on the points inside a sphere of fixed radius. [8] combines several estimates obtained at different sampling rates to compute a more reliable normal. [4] computes the normal as the vector pointing to the centre of the largest Delaunay sphere passing through the considered point. [11] creates several random subsamples of the neighbourhood and uses them to estimate the local noise; then it uses the local noise estimate to compute a kernel function which is minimised by the tangent plane.

A bilateral filter for normal smoothing was proposed in [9]. There, the influence of the normal at d_j on the normal at d_i depends on the distance between d_j and d_j and also on the distance from d_i to the tangent plane at d_j. Most of the point set smoothing algorithms proposed in the literature focus on updating the positions of the points, while the normals are only implicitly updated. [15] uses a filter in the spatial domain, [14] filters in the frequency domain, [10] uses anisotropic diffusion, while [17] optimises simultaneously the point set and its parametrisation by minimising an energy function.

Bootstrap methods are rarely used in geometric modelling problems, probably because they are considered computationally expensive. [3] uses bootstrap to fit ellipses on 2D data. [1] uses bootstrap for polynomial surface fitting with overfitting control. The ensemble technique, which similarly to the bootstrap generates and processes several random subsamples of the input data, was used in [16] for normal reconstruction.

Tackling a problem similar to ours, [2] computes confidence values for the normals of a point set and then uses them to find neighbourhoods of optimal size. However, the computations there rely on the knowledge of scanner parameters, while our input is an unorganised point set only.

1.2 Overview

The rest of the paper is organised as follows. Section 2 describes the proposed bootstrap-based method for normal reconstruction. Section 3 presents experimental results supporting our claim that bootstrap variance can be used as confidence value. Section 4 uses the bootstrap variance to construct a bilateral filter for normal smoothing and Section 5 briefly concludes the paper.

2 Bootstrap Methods for Normal Reconstruction

We first give a brief general description of bootstrap methods, following mainly [6]. Let $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$ be a set of input data. We randomly sample $\mathcal{D}$ with replacement to draw a bootstrap sample S_b with the same size as $\mathcal{D}$. As we sample with replacement, the expected number of distinct elements in S_b is lower than N. In fact, we can easily calculate that the expectation for the number of distinct elements in S_b is

$$N \cdot (1 - \frac{1}{e}) \approx N \cdot 0.632 \tag{1}$$

The sampling procedure is repeated B times and the bootstrap samples

$$S_1, S_2, \ldots, S_B \tag{2}$$

are generated. In our experiments, we typically use $B = 50$.

If $f(\mathcal{D})$ is any quantity computed from the data $\mathcal{D}$, we can use the bootstrap samples in (2) to estimate any aspect of the distribution of $f(\mathcal{D})$. In particular we can estimate its mean by

$$\bar{f}(\mathcal{D}) = \frac{1}{B} \sum_{b=1}^{B} f(S_b) \tag{3}$$

and its variance by

$$\widehat{Var}[f(\mathcal{D})] = \frac{1}{B-1} \sum_{b=1}^{B} (f(S_b) - \bar{f}(\mathcal{D}))^2. \tag{4}$$

Given a sufficiently large number of bootstrap samples B, the bootstrap statistics in (3) and (4) are good approximations of the true mean and variance of $f(\mathcal{D})$ because the distribution of the $f(S_b)$'s follows the distribution of $f(\mathcal{D})$ under non-informative priors, see [6].

2.1 Normal Reconstruction

Applying the bootstrap method to the problem of normal reconstruction, let the input of the algorithm be an unorganised 3D point set $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$. For a given point $d_i \in \mathcal{D}$ we find its k-neighbourhood

$$N_i = \{p_1, p_2, \ldots, p_k\}, \quad p_i \in \mathcal{D}, \quad i = 1, 2, \ldots, k. \tag{5}$$

Following the bootstrap sampling procedure, we generate a set of bootstrap samples $\{S_1, S_2, \ldots, S_B\}$ consisting of B subsets of N_i. On each of the S_i's we apply PCA and compute a tangent plane. For a consistent orientation of the normals of these planes, we first apply PCA on the whole dataset $\mathcal{D}$ and by choosing an arbitrary orientation we compute a normal $\mathbf{n}_{\mathcal{D}}$. Then, we obtain the bootstrap normals $\{\mathbf{n}_1, \mathbf{n}_2, \ldots, \mathbf{n}_B\}$ by choosing for $\mathbf{n}_i$ the orientation that minimises the angle between $\mathbf{n}_{\mathcal{D}}$ and $\mathbf{n}_i$.

Our final estimate for the normal at d_i is the average of the bootstrap normals computed on N_i:

$$\mathbf{n}(d_i) = (\sum_{b=1}^{B} \mathbf{n}_b)/|\sum_{b=1}^{B} \mathbf{n}_b|. \tag{6}$$

This statistic can be seen as the result of (3), where the function f defined on the subsets of N_i returns the PCA normal of a subset.

The second bootstrap statistic we compute on d_i is the variance of the angles between the bootstrap normals and the normal of the first bootstrap sample. That is, we compute the variance

$$\nu_i = \widehat{Var}\{< \mathbf{n}_1, \mathbf{n}_i > |i = 1, \ldots, B\} \tag{7}$$

whereas $< \mathbf{n}_1, \mathbf{n}_i >$ is the angle between $\mathbf{n}_1$ and $\mathbf{n}_i$. This statistic can be seen as the result of (4), where the function f defined on the subsets of N_i returns the angle between the PCA normal of a subset and $\mathbf{n}_1$.

This second statistic will be used as a confidence value for our normal estimate $\mathbf{n}(d_i)$. Notice that other bootstrap statistics could also be treated as confidence values for the normal estimates. In particular, one could treat the PCA tangent planes as the fitted models, compute distances between points in N_i and these tangent planes and use them to compute one of the several bootstrap error estimates proposed in the literature, see [6]. However, we think that processing angles between normal vectors, rather than distances between data points and tangent planes, is a more direct approach and thus more likely to give reliable confidence values.

3 Experimental Results

We tested the proposed normal reconstruction method on synthetic and natural data. Starting from two smooth triangle meshes, the Sphere and the Bunny, we used their connectivity to compute reliable normals at the vertices. These normals were used as the ground truth against which we measure the error of the estimated normals.

Next, we created the test point sets by stripping the mesh connectivity, and as a way of simulating raw data, by adding to each point a random displacement in the direction of the previously computed normals. A noise level equal to γ refers to displacements of magnitude uniformly randomly sampled from the interval $[0, \gamma h]$, where h is the average, all over the model, of the distance between a point and its nearest neighbour. Fig. 1 shows the Sphere model at various levels of noise.

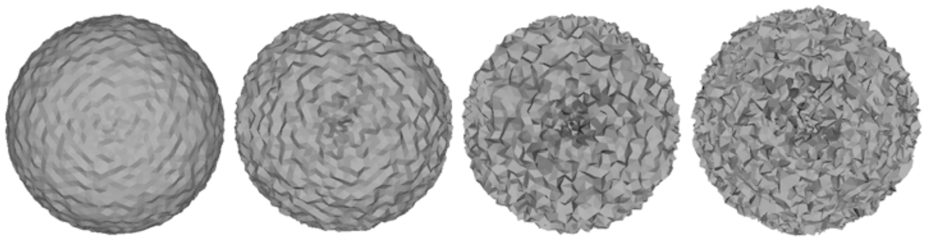


Fig. 1. From left to right: the noise level is 0.25, 0.5, 1.0 and 1.5

3.1 Normal Reconstruction

Table 1 summarises the results of the bootstrap normal reconstruction. The Sphere and the Bunny models were tested at noise levels of 0.25, 0.5, 1.0 and 1.5, and with neighbourhood sizes of 15 and 30. The reported numbers are the average angle differences (in radians) with the ground truth normals. We conclude that the bootstrap normal reconstruction and the PCA normal reconstruction produce comparable results. We also note that the bootstrap method is computationally more intensive, taking 529 seconds to process the 11146 vertices of the Bunny model on a low-end commodity PC.

Table 1. Average angle difference between the ground truth normals and the estimated normals in radians. The number of bootstrap samples is always $B = 50$.

Sphere	k=15		k=30	
Noise	PCA	Bootst.	PCA	Bootst.
0.25	0.0546	0.0547	0.0296	0.0297
0.50	0.1141	0.1151	0.0586	0.0591
1.00	0.2818	0.2847	0.1306	0.1311
1.50	0.4881	0.4949	0.2601	0.2599

Bunny	k=15		k=30	
Noise	PCA	Bootst.	PCA	Bootst.
0.25	0.1388	0.1397	0.1732	0.1739
0.50	0.1754	0.1748	0.1880	0.1882
1.00	0.3175	0.3152	0.2471	0.2463
1.50	0.4660	0.4670	0.3278	0.3264

3.2 The Normal Orientation Problem

The bootstrap normals are computed by (6) as averages of B normals. As PCA gives unoriented vectors, the orientations of these B normals were computed separately. Table 2 shows the error of the bootstrap normal estimation when different normal orientation methods are used.

The columns under Bootst.(1) show the results when an orientation that is consistent with the ground truth normal is selected. In this case, the bootstrap normal estimation outperforms PCA. Of course, as the ground truth normal is not known, the method cannot be used for normal estimation, however, the result shows that bootstrap normal estimation combined with an accurate

Table 2. Average angle difference between the ground truth normals and the estimated normals in radians. In Bootst.(1), the normals generated from the bootstrap samples were oriented consistently with the ground truth normal. In Bootst.(2) the orientation was random. The number of bootstrap samples was always $B = 50$.

Sphere	k=15			k=30		
Noise	PCA	Bootst.(1)	Bootst.(2)	PCA	Bootst.(1)	Bootst.(2)
0.25	0.0546	0.0545	0.1644	0.0296	0.0296	0.1403
0.50	0.1141	0.1137	0.2348	0.0586	0.0591	0.1664
1.00	0.2818	0.2456	0.4633	0.1306	0.1290	0.2665
1.50	0.4881	0.3752	0.6582	0.2601	0.2242	0.4178

Bunny	k=15			k=30		
Noise	PCA	Bootst.(1)	Bootst.(2)	PCA	Bootst.(1)	Bootst.(2)
0.25	0.1388	0.1370	0.2100	0.1732	0.1696	0.2331
0.50	0.1754	0.1664	0.2600	0.1880	0.1817	0.2547
1.00	0.3175	0.2776	0.4172	0.2471	0.2330	0.3206
1.50	0.4660	0.3826	0.5781	0.3278	0.2970	0.4066

vector orientation algorithm could potentially outperform PCA. The columns under Bootst.(2) show the results when a random orientation is chosen for each normal. In this case the results are clearly worse than PCA, showing that a good normal orientation algorithm is essential for accurate bootstrap normal estimations.

Finally, we notice that the consistent orientation of a set of normal estimates corresponding to the same data point is a problem that has attracted considerably less research interest than the problem of consistent normal orientation over a point set, see for example [7].

3.3 Bootstrap Variance

In a second experiment, we examine the claim that normals with higher bootstrap variance are generally less accurate. Working on the Bunny and Fandisk models at various noise levels, we split the set of vertices into a high variance subset, containing the vertices with the highest 15% variances, and a low variance subset, containing the rest of the vertices. We compute and report the average normal error of the two sets separately. The results are summarised in Table 3.

Table 3. The average normal error on the high and low variance subsets of the Bunny and the Fandisk models

Bunny	0.25	0.50	1.00	1.50
Low Var.	0.1707	0.1768	0.2082	0.2489
High Var.	0.4731	0.4920	0.5168	0.5927

Fandisk	0.25	0.50	1.00	1.50
Low Var.	0.0590	0.0745	0.1146	0.1818
High Var.	0.5813	0.5746	0.5850	0.5911

We notice that, as expected, the high variance set has normal estimates with significantly higher average error. Moreover, this is the case at all noise levels, meaning that the method can handle error from both sources, that is, the model features and the added noise. As a limitation of the approach, we note that in all comparisons we used the same neighbourhood size of $k = 50$ because higher or lower values of k would respectively naturally decrease or increase the variance, without necessarily changing the accuracy of the normal estimates.

Fig. 2 shows colourmaps of the bootstrap variance for the Bunny and Fandisk models at various levels of noise. We notice that the high variance areas concentrated on the features of the mesh. We also notice that this pattern degrades slowly as the added noise increases.

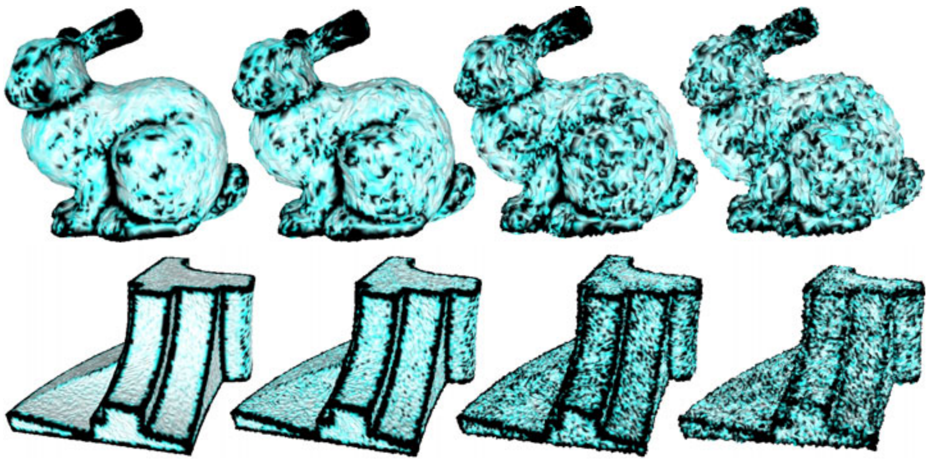


Fig. 2. From left to right: Colourmaps of the bootstrap variance at noise levels 0.25, 0.5, 1.0 and 1.5, respectively. The darker colours signify higher variance.

4 Bilateral Gaussian Filter for Normal Smoothing

In Section 3 we showed that normal estimates with lower bootstrap variance are generally more reliable than those with higher variance. In this section, we will use the bootstrap variance as a confidence value and propose a bilateral Gaussian filter for normal smoothing. In each filtering iteration, the normal at a point d_i is updated as a distance weighted average of the normals at the neighborhood of d_i, while the confidence values are used to reduce the influence of the less reliable normals.

4.1 Bilateral Gaussian Filter

One iteration of the proposed filter updates the normal $\mathbf{n}(d_i)$ at a data point $d_i \in \mathcal{D}$ by

$$\mathbf{n}(d_i) \rightarrow (1-\alpha)\mathbf{n}(d_i) + \frac{\alpha}{c(d_i)} \sum_{p \in N_i} \mathcal{G}(\nu_j, \sigma_\nu)\mathcal{G}(||d_i - p||, \sigma_d)\mathbf{n}(p) \tag{8}$$

In (8), $\mathcal{G}(a,b)$ is the Gaussian function with zero mean $\mathcal{G}(a,b) = e^{-a^2/b^2}$, $||\cdot||$ denotes Euclidean distance, and $c(d_i)$ is a normalisation factor that makes the sum of the weights equal to 1, that is

$$c(d_i) = \sum_{p \in N_i} \mathcal{G}(\nu_j, \sigma_\nu)\mathcal{G}(||d_i - p||, \sigma_d) \tag{9}$$

The variances σ_ν and σ_d of the two Gaussians in (8) are user defined parameters. A smaller value of σ_ν would mean normals with high variance, that is, less reliable normals, will have smaller influenced on the smoothing process. A smaller value of σ_d would mean that the influence of a point decays more rapidly with the distance. In our experiments, we used a σ_d equal to the average, all over the model, of the distance between a point and its nearest neighbour. Finally, α is a user defined parameter controlling the strength of the filtering operation.

4.2 Evaluation

We tested the algorithm on the Cube and the Fandisk point sets with 0.5 level of added noise, $\sigma_\nu = 10^{-12}$ and 1.1×10^{-12}, respectively, and $\alpha = 0.1$. As in Section 3, we used the normals obtained from the underlying meshes before the addition of noise as the ground truth.

The results and a comparison with the single Gaussian filter corresponding to $\sigma_\nu = \infty$ are shown in Fig. 3. The computed normal errors are shown separately for the feature areas where $\nu_i \geq 0.04$ (left), and the non-feature areas (right). In all cases, the proposed bilateral filter outperforms the single Gaussian filter. We also notice that in the non-feature areas the normal error increases from the very first iteration of the single Gaussian filter. The reason is that the bad normal estimates in the feature areas corrupt the more reliable estimates in the smoother areas. This is a serious limitation of the single Gaussian filter, which is overcome by the proposed bilateral filter.

Fig. 4 shows the normals of the Cube model after applying the bilateral filter and after applying the simple Gaussian filter. We conclude that the superiority of the bilateral filter, as demonstrated by the graphs in Fig. 3 (top), is visually significant.

Notice that the above validation of the filtering algorithm is mainly concerned with the accuracy of the bootstrap normals compared to a ground truth, here the normals obtained from a smooth triangle mesh whose vertices were used in the construction of the test data sets. On the other hand, as one of the primary uses of point sets is the fast, high quality rendering of 3D models, qualitative, visual evaluations of normal smoothing algorithms are also common in the literature, see for example [9].

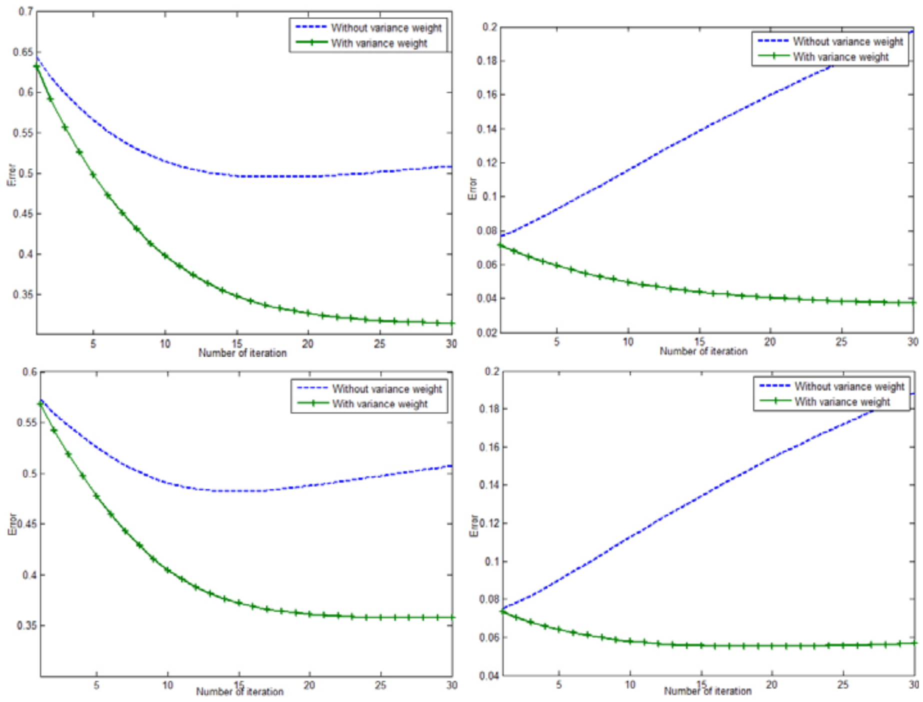


Fig. 3. Comparison between the single Gaussian filter (blue dotted line) and the proposed bilateral filter (green crossed line). **Top:** Cube with 0.5 noise level. **Bottom:** Fandisk with 0.5 noise level. **Left:** Feature areas. **Right:** Non-feature areas.

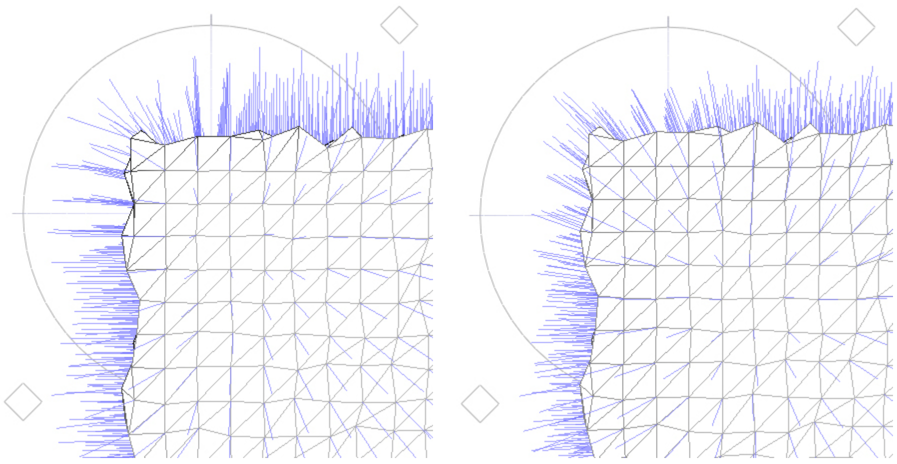


Fig. 4. Left: The proposed bilateral filter. **Right:** Simple Gaussian filter. In both figures, the angle of view is centered at the circle.

5 Conclusion

We proposed a bootstrap-based method for normal reconstruction. The experimental evaluation of the proposed method showed that although it does not significantly improve the accuracy of the normal estimates, it can provide reliable confidence values for them. The utility of such confidence values was demonstrated by the construction of a bilateral Gaussian filter for normal smoothing.

In the future, we plan to investigate further applications of the confidence values of the normal estimates. In particular, we plan to use the information about the quality of the normals to adaptively select optimally sizes neighbourhoods for normal estimation. We notice that in the flat parts of the surface simple PCA gives reliable estimates, while in the high curvature areas normal estimation is an inherently ill-posed problem. Thus, the focus will be on improving the normal estimations in the in between areas, that is, flat areas close to high curvature areas. The main challenge will be to choose neighborhoods that will be sufficiently small not to include high curvature points, but sufficiently large to give stable normal estimates.

References

1. Ramli, A., Ivrissimtzis, I.: Bootstrap test error estimations of polynomial fittings in surface reconstruction. In: Proceedings of VMV, pp. 101–112 (2009)
2. Bae, K.-H., Belton, D., Lichti, D.D.: A closed-form expression of the positional uncertainty for 3d point clouds. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(4), 577–590 (2009)
3. Cabrera, J., Meer, P.: Unbiased estimation of ellipses by bootstrapping. IEEE Transactions on Pattern Analysis and Machine Intelligence 18(7), 752–756 (1996)
4. Dey, T.K., Li, G., Sun, J.: Normal estimation for point clouds: A comparison study for a voronoi based method. In: Proceeding of SoPBG, pp. 39–46 (2005)
5. Gopi, M., Krishnan, S., Silva, C.T.: Surface reconstruction based on lower dimensional localized Delaunay triangulation. Computer Graphics Forum 19(3), 467–478 (2000)
6. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, Heidelberg (2001)
7. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points. In: Proceedings of SIGGRAPH, pp. 71–78 (1992)
8. Hu, G., Xu, J., Miao, L., Peng, Q.-S.: Bilateral estimation of vertex normal for point-sampled models. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3480, pp. 758–768. Springer, Heidelberg (2005)
9. Jones, T.R., Durand, F., Zwicker, M.: Normal improvement for point rendering. IEEE Computer Graphics and Applications 24(4), 53–56 (2004)
10. Lange, C., Polthier, K.: Anisotropic smoothing of point sets. Computer Aided Geometric Design 22(7), 680–692 (2005)
11. Li, B., Schnabel, R., Klein, R., Cheng, Z., Dang, G., Jin, S.: Robust normal estimation for point clouds with sharp features. Computers & Graphics 34(2), 94–106 (2010)

12. Mitra, N.J., Nguyen, A., Guibas, L.: Estimating surface normals in noisy point cloud data. Special Issue of International Journal of Computational Geometry and Applications 14(4-5), 261–276 (2004)
13. Pauly, M., Keiser, R., Kobbelt, L., Gross, M.: Shape modeling with point-sampled geometry. ACM Transactions on Graphics 22(3), 641–650 (2003)
14. Pauly, M., Kobbelt, L.P., Gross, M.: Point-based multiscale surface representation. ACM Transactions on Graphics 25(2), 177–193 (2006)
15. Qin, H., Yang, J., Zhu, Y.: Nonuniform bilateral filtering for point sets and surface attributes. The Visual Computer 24, 1067–1074 (2008)
16. Yoon, M., Lee, Y., Lee, S., Ivrissimtzis, I., Seidel, H.-P.: Surface and normal ensembles for surface reconstruction. Computer Aided Design 39(5), 408–420 (2007)
17. Zhang, L., Liu, L., Gotsman, C., Huang, H.: Mesh reconstruction by meshless denoising and parameterization. Computers & Graphics 34(3), 198–208 (2010)

Couple Points – A Local Approach to Global Surface Analysis

Christian Rössl and Holger Theisel

Visual Computing Group, University of Magdeburg, Germany

Abstract. We introduce the concept of couple points as a global feature of surfaces. Couple points are pairs of points $(\mathbf{x}_1, \mathbf{x}_2)$ on a surface with the property that the vector $\mathbf{x}_2 - \mathbf{x}_1$ is parallel to the surface normals both at $\mathbf{x}_1$ and $\mathbf{x}_2$. In order to detect and classify them, we use higher order local feature detection methods, namely a Morse theoretic approach on a 4D scalar field. We apply couple points to a number of problems in Computer Graphics: the detection of maximal and minimal distances of surfaces, a fast approximation of the shortest geodesic path between two surface points, and the creation of stabilizing connections of a surface.

Keywords: surface features, double normals, Morse theory, triangular meshes, geodesic paths.

1 Introduction

Size, complexity and number of surfaces considered in Computer Graphics are continuously growing. One popular approach to deal with this is the extraction of characteristic features of a surface. Feature extraction has a variety of applications such as segmentation, shape matching, reverse engineering, compression and simplification of surfaces.

Generally, two kinds of surface features can be distinguished: local and global features. For local features it can be decided entirely by a local analysis whether or not a point on the surface belongs to the feature. For global features, this decision can be made only by a global analysis of the surface.

Examples for detecting local features on surfaces are the estimation of the curvature tensor (see, e.g., [1,8,19,22] or the survey [11]), the estimation of surface normals, or the detection of sharp edges. Examples for global features are the detection of medial axes [3,23], the shortest distance between a point and a surface [9], and the detection of the shortest geodesic path between two points on a surface. This is an interesting and well-studied problem which leads to solving the associated PDE by propagating wave fronts [15] or to unfolding a polyhedral surface [2,24,26] (see also [17] for a general survey). While such algorithms are efficient for single-source approximations of shortest paths, they may be rather expensive if paths between arbitrary point pairs have to be computed. Alternative methods apply energy (arc-length) minimization [10,13,20] for a curve in a manifold, where a reasonable initial guess is required, e.g., from searching a discrete shortest path. In contrast to the point-point problem which imposes a boundary value problem, the construction of a straightest geodesic path starting from a point and proceeding into a given direction is much simpler: such initial value problem depends entirely on a local analysis [21].

A combination of local and global features is the extraction of Morse complexes on a surface [4,5,18]. Given a smooth function m on a surface, Morse complexes divide the surface into areas of similar behavior of the gradient flow of m. To do so, critical points — i.e., points with a vanishing gradient of m — are extracted and classified into sources, sinks, and saddles. Then certain separation curves are integrated from the saddles in forward and backward gradient direction. Once the scalar field m is given, the extraction of the critical points is a local process. However, the integrated separation lines reveal a global feature: local changes of m can cause potential changes of the separation lines everywhere on the surface. Also note that the underlying Morse function may be obtained by a local or a global [12] analysis of the surface.

This paper introduces a new global feature of the surface called couple points. These are pairs of points $(\mathbf{x}_1, \mathbf{x}_2)$ on the surface with the property that the vector $\mathbf{x}_2 - \mathbf{x}_1$ is parallel to the surface normals both at $\mathbf{x}_1$ and $\mathbf{x}_2$. In order to extract and classify them, we do not apply a global analysis of the surface but a local feature analysis on pairs of surfaces instead. This way we are able to apply standard local analysis tools to achieve a global analysis of the surfaces. We show that in general there exists a finite number of isolated couple points. We believe that number, location, and classification of couple points reveals a relevant information about the global shape of surfaces.

We remark that the general concept of couple points is not new. In fact, the definition of double normals is equivalent to couple points, and there a number of interesting problems in geometry related to them, e.g., finding their minimum number for convex bodies in E^n [14] or for smooth curves [7] of certain topology. We are interested in the detection of couple points and applications in computer graphics.

The remainder of this article paper is organized as follows: Section 2 gives the definition of couple points and shows a first simple example. Section 3 collects properties of couple points on smooth surfaces. To do so, we apply Morse theoretic approaches to an appropriate 4D scalar field and show that the couple points correspond to the critical points of this 4D Morse complex. Section 4 shows how to extract and classify couple points on triangular meshes. In Section 5 we apply couple points to the following problems: first, we detect minimal (or maximal) distances between two surfaces (a problem which is far more complicated than the problem of finding the shortest distance between a point and a surface). Second, we use couple points to compute a fast approximation of the shortest geodesic path between two points on a surface. And third, we use couple points to find stabilizing connectors on surfaces in order to increase the surface stability. Finally, Section 6 draws conclusions and mentions future research.

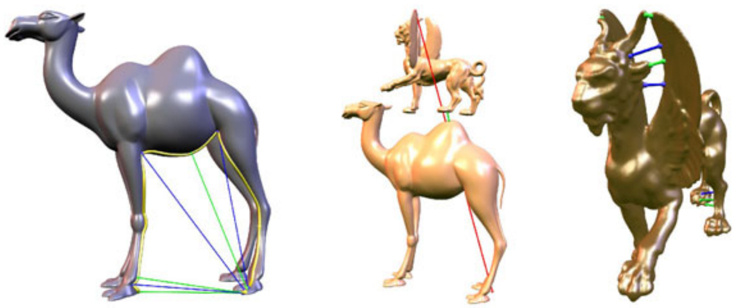


Fig. 1. Three applications of couple points. Left: a fast approximation of the shortest path between two surface points (yellow) passing trough five couple points. Middle: shortest (green) and largest (red) distance between two surfaces. Right: stabilizing connectors between parts of a surface.

2 Motivation and Definition of Couple Points

To introduce the idea of couple points, we start with a simple 2D example. Given are two closed differentiable curves $\mathbf{x}_1, \mathbf{x}_2$ which do not intersect each other. We search for the minimal and maximal Euclidean distance of $\mathbf{x}_1$ and $\mathbf{x}_2$, i.e., for a pair of points $(\mathbf{x}_1^{min}, \mathbf{x}_2^{min}) \in (\mathbf{x}_1, \mathbf{x}_2)$ and $(\mathbf{x}_1^{max}, \mathbf{x}_2^{max}) \in (\mathbf{x}_1, \mathbf{x}_2)$ with

$$\forall (\mathbf{x}_1^i, \mathbf{x}_2^i) \in (\mathbf{x}_1, \mathbf{x}_2) \quad : \quad \|\mathbf{x}_2^{min} - \mathbf{x}_1^{min}\| \leq \|\mathbf{x}_2^i - \mathbf{x}_1^i\| \leq \|\mathbf{x}_2^{max} - \mathbf{x}_1^{max}\|.$$

Figure 2a gives an illustration. In this picture we can also observe a property of

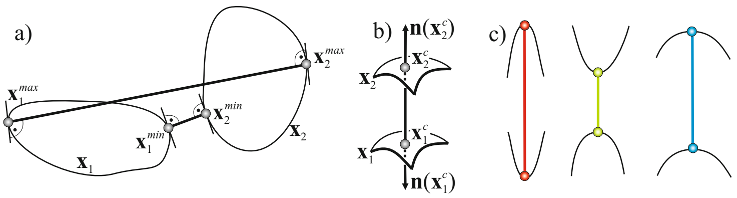


Fig. 2. a) Pair of points $(\mathbf{x}_1^{min}, \mathbf{x}_2^{min})$ and $(\mathbf{x}_1^{max}, \mathbf{x}_2^{max})$ with minimal and maximal distance. b) A couple point. c) Classification of couple points in sink (red), source (green) and saddle (blue).

$(\mathbf{x}_1^{min}, \mathbf{x}_2^{min})$ (which will be proved later in Section 3): the vector $\mathbf{x}_2^{min} - \mathbf{x}_1^{min}$ is perpendicular to the tangent direction of $\mathbf{x}_1$ in $\mathbf{x}_1^{min}$ and of $\mathbf{x}_2$ in $\mathbf{x}_2^{min}$. A similar statement holds for $(\mathbf{x}_1^{max}, \mathbf{x}_2^{max})$. The transformation of this observation to surfaces gives reason for the following definition of couple points:.

Definition 1. *Given two differentiable surfaces* $\mathbf{x}_1, \mathbf{x}_2$ *together with their normal maps* $\mathbf{n}(\mathbf{x}_1), \mathbf{n}(\mathbf{x}_2)$*, a* couple point $\mathbf{x}^c = (\mathbf{x}_1^c, \mathbf{x}_2^c)$ *is a pair of points with*

$\mathbf{x}_1^c \in \mathbf{x}_1$, $\mathbf{x}_2^c \in \mathbf{x}_2$, *and* $\mathbf{n}(\mathbf{x}_1^c) \times (\mathbf{x}_2^c - \mathbf{x}_1^c) = \mathbf{n}(\mathbf{x}_2^c) \times (\mathbf{x}_2^c - \mathbf{x}_1^c) = (0,0,0)^T$. *Furthermore, let* $C(\mathbf{x}_1, \mathbf{x}_2)$ *be the set of all couple points between* $\mathbf{x}_1$ *and* $\mathbf{x}_2$.

Figure 2b illustrates a couple point. Given two surfaces $\mathbf{x}_1$ and $\mathbf{x}_2$, there is in general a finite number of isolated couple points between them. This paper is devoted to studying their properties and their applicability to a number of problems in Computer Graphics. Couple points can also be computed on a single surface, i.e., for instance $C(\mathbf{x}_1, \mathbf{x}_1)$ can be extracted.

3 Properties of Couple Points

In order to capture useful properties of couple points, we first assume $\mathbf{x}_1$ and $\mathbf{x}_2$ to be parametric surfaces. Then we apply a Morse theoretical approach to the 4D domain of $\mathbf{x}_1$ and $\mathbf{x}_2$ and show that our derived properties of couple points are independent of the particular parametrization of the surfaces.

Given two regularly parametrized surfaces $\mathbf{x}_1(u,v)$ over a 2D domain D_1 and $\mathbf{x}_2(s,t)$ over a 2D domain D_2, we define a *double surface* $\mathbf{x}^d$ as

$$\mathbf{x}^d(u,v,s,t) = (\mathbf{x}_1(u,v), \mathbf{x}_2(s,t)). \tag{1}$$

This means that $\mathbf{x}^d$ is a map from $D = D_1 \times D_2$ to $\mathbb{R}^3 \times \mathbb{R}^3$. A point on a double surface is called a *double point*. Furthermore, we define the *double normal*

$$\mathbf{n}^d(u,v,s,t) = (\mathbf{n}_1, \mathbf{n}_2) = \left(\frac{\mathbf{x}_{1u} \times \mathbf{x}_{1v}}{\|\mathbf{x}_{1u} \times \mathbf{x}_{1v}\|} \; , \; \frac{\mathbf{x}_{2s} \times \mathbf{x}_{2t}}{\|\mathbf{x}_{2s} \times \mathbf{x}_{2t}\|} \right). \tag{2}$$

where $\mathbf{x}_{1u}, \mathbf{x}_{1v}, \mathbf{x}_{2s}, \mathbf{x}_{2t}$ are the first order partials of $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively. We consider the 4D Morse function

$$m(u,v,s,t) = \|\mathbf{x}_2(u,v) - \mathbf{x}_1(s,t)\|^2 \tag{3}$$

which describes the (squared) Euclidean distance of $\mathbf{x}_1$ and $\mathbf{x}_2$. Its gradient $\mathrm{grad}(m) = (m_u, m_v, m_s, m_t)$ is a 4D vector field on D, its map onto $\mathbf{x}^d$ gives the *gradient double vector*

$$\begin{aligned} \mathbf{v}^d(u,v,s,t) &= (\mathbf{v}_1(u,v,s,t) \; , \; \mathbf{v}_2(u,v,s,t)) \qquad\qquad (4) \\ &= \Bigg(m_u \frac{\mathbf{x}_{1v} \times \mathbf{n}_1}{\|\mathbf{x}_{1u} \times \mathbf{x}_{1v}\|} + m_v \frac{\mathbf{n}_1 \times \mathbf{x}_{1u}}{\|\mathbf{x}_{1u} \times \mathbf{x}_{1v}\|} \; , \\ &\qquad m_s \frac{\mathbf{x}_{2t} \times \mathbf{n}_2}{\|\mathbf{x}_{2s} \times \mathbf{x}_{2t}\|} + m_t \frac{\mathbf{n}_2 \times \mathbf{x}_{2t}}{\|\mathbf{x}_{2s} \times \mathbf{x}_{2t}\|} \Bigg). \end{aligned}$$

$\mathbf{v}^d$ points into the direction of steepest ascent of the Euclidean distance of $\mathbf{x}_1$ and $\mathbf{x}_2$. Figure 3a illustrates $\mathbf{x}^d$, $\mathbf{n}^d$ and $\mathbf{v}^d$.

The following theorem will provide the foundation of our further treatment of couple points.

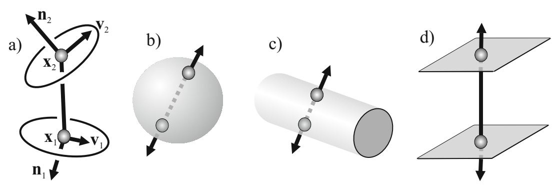


Fig. 3. a) A double point $\mathbf{x}^d = (\mathbf{x}_1, \mathbf{x}_2)$ with double normal $\mathbf{n}^d = (\mathbf{n}_1, \mathbf{n}_2)$ and gradient double vector $\mathbf{v}^d = (\mathbf{v}_1, \mathbf{v}_2)$. b)-e) Zero importance couple points: diametric on a sphere, opposite on a cylinder, opposite on parallel planes.

Theorem 1. *Given $\mathbf{x}^d$, $\mathbf{n}^d$, $\mathbf{v}^d$ as defined in (1) - (4), the following equation holds.*

$$\mathbf{v}^d = (\ 2((\mathbf{x}_2 - \mathbf{x}_1) \times \mathbf{n}_1) \times \mathbf{n}_1\ \ ,\ \ 2((\mathbf{x}_1 - \mathbf{x}_2) \times \mathbf{n}_2) \times \mathbf{n}_2\ )\ .$$

The proof is a straightforward exercise in algebra from (1)-(4). Theorem 1 shows that $\mathbf{v}$ is a geometric measure on $\mathbf{x}_1$ and $\mathbf{x}_2$, i.e., it is independent of the particular parametrization of $\mathbf{x}_1$ and $\mathbf{x}_2$. In fact, $\mathbf{v}^d$ can directly be computed from $\mathbf{x}^d$ and $\mathbf{n}^d$. Also from Theorem 1 follows directly

Theorem 2. *Given $\mathbf{x}^d$, $\mathbf{n}^d$, $\mathbf{v}^d$ as defined from (1) - (4), the following three statements are equivalent:*

- $\mathbf{x}^d(u,v,s,t)$ *is a couple point concerning definition 1.*
- *grad*$(m) = (0,0,0,0)^T$.
- $\mathbf{v}^d = ((0,0,0)^T, (0,0,0)^T) = \mathbf{0}^d$, *i.e.* $\mathbf{x}^d$ *is a critical (double) point w.r.t.* $\mathbf{v}^d$.

Let $\mathbf{x}^c = \mathbf{x}^d(u_c, v_c, s_c, t_c)$ be a couple point. We apply a local reparametrization of $\mathbf{x}_1$ and $\mathbf{x}_2$ such that $\mathbf{x}_{1u}(u_c, v_c)$ and $\mathbf{x}_{1v}(u_c, v_c)$ are orthonormalized, and that $\mathbf{x}_{2s}(u_c, v_c)$ and $\mathbf{x}_{2t}(u_c, v_c)$ are orthonormalized as well. This can easily be done by locally computing normal and principal directions and using these vectors as the bases of a local coordinate system. Then we can further classify $\mathbf{x}^c$ by an eigen-analysis of the Hessian matrix

$$H_m(u,v,s,t) = \begin{pmatrix} m_{uu} & m_{uv} & m_{us} & m_{ut} \\ m_{vu} & m_{vv} & m_{vs} & m_{vt} \\ m_{su} & m_{sv} & m_{ss} & m_{st} \\ m_{tu} & m_{tv} & m_{ts} & m_{tt} \end{pmatrix}$$

at (u_c, v_c, s_c, t_c). Let λ_i $(i = 1,..,4)$ be the eigenvalues of $H_m(u_c, v_c, s_c, t_c)$. We call $\mathbf{x}^c$ a *source* iff all eigenvalues of $H_m(u_c, v_c, s_c, t_c)$ are positive. In this case, m has a local minimum at (u_c, v_c, s_c, t_c): all double points on $\mathbf{x}^d$ in a small neighborhood of $\mathbf{x}^c$ have Euclidean distance smaller than $\|\mathbf{x}_2(s_c, t_c) - \mathbf{x}_1(u_c, v_c)\|$. $\mathbf{x}_c$ is a *sink* iff all eigenvalues of H_m are negative, i.e., m has a local maximum at (u_c, v_c, s_c, t_c). $\mathbf{x}^c$ is a *saddle* iff it has both positive and negative eigenvalues. Figure 2c illustrates source, sink, and saddle couple points for 2D curves. There, as well as in the remaining figures, we represent a source couple point with a green line, a sink with a red line, and a saddle with a blue line.

If we consider the couple points of only one surface $\mathbf{x}$ (i.e. we consider $C(\mathbf{x},\mathbf{x})$), and if $\mathbf{x}$ is a closed manifold, then a couple point $\mathbf{x}^c = (\mathbf{x}_1^c, \mathbf{x}_2^c) \in (\mathbf{x},\mathbf{x})$ can be further classified as

- *direct inside* if the line segment $\mathbf{x}_1^c, \mathbf{x}_2^c$ is completely inside the surface.
- *direct outside* if the line segment $\mathbf{x}_1^c, \mathbf{x}_2^c$ is completely outside the surface.
- *indirect* else.

This distinction is useful for applications which connect couple points by a straight line which should not have intersections with the surface, e.g., stabilizing connectors (see Section 5.3). Figure 4 illustrates this for a closed 2D curve.

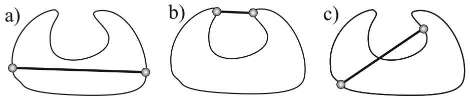


Fig. 4. Classification of couple points: a) direct inside, b) direct outside, c) indirect

As we will see later, there may be a rather large or even infinite number of couple points in a surface. For instance in 2D curves of constant width, e.g., circle or Reuleaux triangle, have infinitely many couple points. To deal with this, i.e., to discard the unimportant ones, we also need to equip a couple point with an importance. Couple points which tend to disappear after slight changes of the surface should have a low importance. Also, couple points which are not isolated should have a zero importance. Here we assume a sufficiently smooth surface and use $\mathrm{imp}(\mathbf{x}^d) = \det(H_m)$ which fulfills the requirements mentioned above. Figure 3b-e shows some examples of non-isolated couple points with a zero importance, i.e., they are degenerate and are not considered here. We remark that this simple definition of importance is a local property. In particular, this local importance is not necessarily stable under perturbation of the shape or its parametrization by adding noise.

Let $\mathbf{e}_i$ $(i = 1,..,4)$ be the eigenvectors of $H_m(u_c, v_c, s_c, t_c)$. Then $\mathbf{e}_i$ can be transformed to double eigenvectors on $\mathbf{x}^d$ by

$$\mathbf{e}_i^c = (\ a_i\ \mathbf{x}_{1u}(u_c, v_c)\ +\ b_i\ \mathbf{x}_{1v}(u_c, v_c)\ ,\ c_i\ \mathbf{x}_{2s}(s_c, t_c)\ +\ d_i\ \mathbf{x}_{2t}(s_c, t_c)\ )$$

for $i = 1,..,4$. Note that $\mathbf{e}_i^c$ are uniquely defined by $\mathbf{x}^c$ and the curvature tensors at $\mathbf{x}_1(u_c, v_c)$ and $\mathbf{x}_2(s_c, t_c)$. Then we can compute the separation double line from $\mathbf{x}_c$ by applying a double stream line integration of $\mathbf{v}^d$ starting in $\mathbf{x}_c$ in the directions $\pm\mathbf{e}_i^c$. The integration direction (either forward or backward) is given by the signs of λ_i. This way, 8 separation double stream lines are created by a couple point $\mathbf{x}^c$. Figure 5 gives an illustration.

The set of all couple points together with all integrated double separation curves gives the topological skeleton of the Morse function. Figure 6 shows an example of a test surface containing 370 couple points. The close-up shows that the separation double curves cover the surface in a rather dense way.

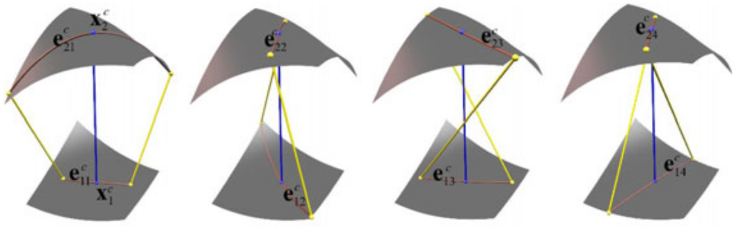


Fig. 5. A couple point $\mathbf{x}^c = (\mathbf{x}_1^c, \mathbf{x}_2^c)$ and its 4 double eigenvectors $\mathbf{e}_i^c = (\mathbf{e}_{1i}^c, \mathbf{e}_{2i}^c)$. Each double eigenvector creates two double stream lines by integrating $\mathbf{v}^d$. Here, they end in double points (yellow) when one of the components reaches the boundary of the surface.

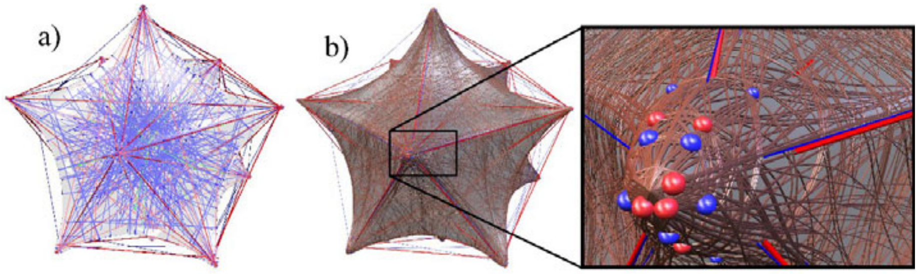


Fig. 6. a) Test surface with 370 couple points. b) All separation double curves.

4 Couple Points for Triangular Meshes

Up to now we treated couple points for smooth surfaces. In this section we show how to apply the concept to piecewise linear approximations of smooth surfaces, i.e. to triangular meshes, as this is the standard surface representation in Computer Graphics. Here, we assume that each vertex is equipped with an either exact or estimated normal. Then the basic approach is to test each pair of triangles for couple points: given a triangle $\mathbf{t}$ with the vertices $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$ and their assigned normals $\mathbf{n}_1$, $\mathbf{n}_2$, $\mathbf{n}_3$, and given a triangle $\widetilde{\mathbf{t}}$ with the vertices $\widetilde{\mathbf{p}_1}$, $\widetilde{\mathbf{p}_2}$, $\widetilde{\mathbf{p}_3}$ and the normals $\widetilde{\mathbf{n}_1}$, $\widetilde{\mathbf{n}_2}$, $\widetilde{\mathbf{n}_3}$, we search for couple points as barycentric combinations (α, β, γ) and $(\widetilde{\alpha}, \widetilde{\beta}, \widetilde{\gamma})$ by solving the system

$$(\widetilde{\mathbf{x}} - \mathbf{x}) \times \mathbf{n} \; = \; (\widetilde{\mathbf{x}} - \mathbf{x}) \times \widetilde{\mathbf{n}} \; = \; (0,0,0)^T$$

with $\mathbf{x} = (\alpha\, \mathbf{p}_1 + \beta\, \mathbf{p}_2 + \gamma\, \mathbf{p}_3)$, $\widetilde{\mathbf{x}} = (\widetilde{\alpha}\, \widetilde{\mathbf{p}_1} + \widetilde{\beta}\, \widetilde{\mathbf{p}_2} + \widetilde{\gamma}\, \widetilde{\mathbf{p}_3})$, $\mathbf{n} = (\alpha\, \mathbf{n}_1 + \beta\, \mathbf{n}_2 + \gamma\, \mathbf{n}_3)$, $\widetilde{\mathbf{n}} = (\widetilde{\alpha}\, \widetilde{\mathbf{n}_1} + \widetilde{\beta}\, \widetilde{\mathbf{n}_2} + \widetilde{\gamma}\, \widetilde{\mathbf{n}_3})$, $\alpha + \beta + \gamma = 1$, $\widetilde{\alpha} + \widetilde{\beta} + \widetilde{\gamma} = 1$ for the unknowns $(\alpha, \beta, \gamma, \widetilde{\alpha}, \widetilde{\beta}, \widetilde{\gamma})$. This ends up in finding the roots of a 6th order polynomial if the intersection line of the two triangle planes is excluded as a solution. This in turn means that two triangles which do not intersect each other can have up to 6 isolated couple points. Figure 7a gives an illustration.

In order to find the couple points between two triangles, we apply a subdivision approach. We test whether or not $\mathbf{t}$ can "see" $\widetilde{\mathbf{t}}$, i.e., whether there are

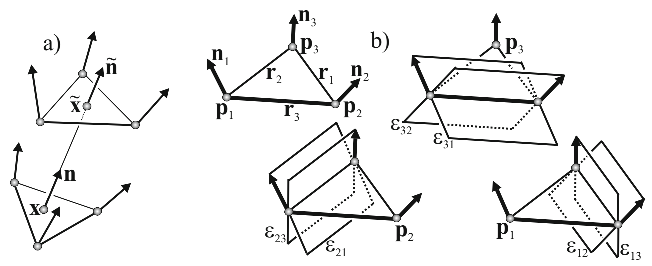


Fig. 7. a) A couple point between two triangles. b) The planes ϵ_{ij}.

barycentric coordinates (α, β, γ) such that the line $\mathbf{x} + \mu\mathbf{n}$ intersects $\widetilde{\mathbf{t}}$. Therefore, we introduce the 6 planes ϵ_{ij} $(i, j \in \{1, 2, 3\}, i \neq j)$ by demanding that ϵ_{ij} contains the line $\mathbf{r}_i$ and the vector $\mathbf{n}_i$. Here, $\mathbf{r}_1$ is the line through $\mathbf{p}_2$ and $\mathbf{p}_3$, $\mathbf{r}_2$ passes through $\mathbf{p}_3$ and $\mathbf{p}_1$, and $\mathbf{r}_3$ passes through $\mathbf{p}_1$ and $\mathbf{p}_2$. Figure 7b illustrates the planes ϵ_{ij}. Now the test is done by checking on which side of the planes the vertices $\widetilde{\mathbf{p}_1}$, $\widetilde{\mathbf{p}_2}$, $\widetilde{\mathbf{p}_3}$ are located. We use the notation $\mathrm{opp}(\mathbf{p}, \epsilon, \mathbf{q})$ if the points $\mathbf{p}$ and $\mathbf{q}$ are located on opposite sides of the plane ϵ. Then we check the following conditions:

- If $\mathrm{opp}(\widetilde{\mathbf{p}}_i, \epsilon_{12}, \mathbf{p}_1) \wedge \mathrm{opp}(\widetilde{\mathbf{p}}_i, \epsilon_{13}, \mathbf{p}_1)$ for all $i \in \{1, 2, 3\}$, then $\mathbf{t}$ cannot see $\widetilde{\mathbf{t}}$.
- If $\mathrm{opp}(\widetilde{\mathbf{p}}_i, \epsilon_{21}, \mathbf{p}_2) \wedge \mathrm{opp}(\widetilde{\mathbf{p}}_i, \epsilon_{23}, \mathbf{p}_2)$ for all $i \in \{1, 2, 3\}$, then $\mathbf{t}$ cannot see $\widetilde{\mathbf{t}}$.
- If $\mathrm{opp}(\widetilde{\mathbf{p}}_i, \epsilon_{31}, \mathbf{p}_3) \wedge \mathrm{opp}(\widetilde{\mathbf{p}}_i, \epsilon_{32}, \mathbf{p}_3)$ for all $i \in \{1, 2, 3\}$, then $\mathbf{t}$ cannot see $\widetilde{\mathbf{t}}$.

If none of these three conditions applies, we assume that $\mathbf{t}$ can "see" $\widetilde{\mathbf{t}}$. In a similar way we check whether $\widetilde{\mathbf{t}}$ can "see" $\mathbf{t}$. If either $\mathbf{t}$ cannot "see" $\widetilde{\mathbf{t}}$ or $\widetilde{\mathbf{t}}$ cannot "see" $\mathbf{t}$, no couple point exists between $\mathbf{t}$ and $\widetilde{\mathbf{t}}$. Otherwise we subdivide $\mathbf{t}$ and $\widetilde{\mathbf{t}}$ and apply our test recursively again until the size of $\mathbf{t}$ and $\widetilde{\mathbf{t}}$ is beyond a certain accuracy threshold.

As presented above, our algorithm for finding all couple points tests every pair of triangles for couple points and consequently has quadratic complexity in terms of the number of triangles. At this point, (hierarchical) space partition techniques can be applied to drastically improve on efficiency. Even simpler is a partition of normal directions which allows the fast enumeration of all candidates which eventually "see" a given triangle.

Once we have detected all couple points of a mesh, we are left with classifying them as sources, sinks, and saddles as well as to compute their eigenvalues and eigenvectors. All we need to have is an estimation of the curvature tensors of the surfaces at $\mathbf{x}_{1c}$ and $\mathbf{x}_{2c}$. Most existing algorithms to estimate the curvature tensor on a mesh do so per vertex. We follow [25] to estimate the curvature tensor at every inner point of a triangle as a smooth function over the triangle. This is done by considering both the linear interpolation of the vertices and the normals. This allows us to compute a classification of couple points as described in Section 3.

5 Results and Applications

In this section we demonstrate the extraction of couple points to a number of test data sets. Then we describe three areas of applications for couple points.

For the camel data set in Figure 8 (consisting of 78,144 triangles) we detected

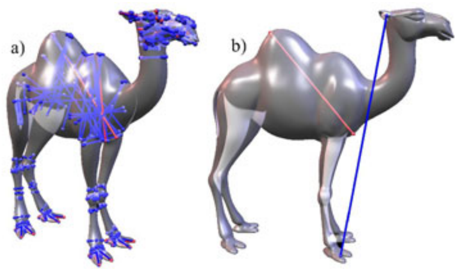


Fig. 8. Camel: a) all direct inside couple points, b) longest direct inside and direct outside couple points

2,721 couple points. Figure 8a shows all detected direct inside couple points. In Figure 8b, we picked two particular couple points: the longest one (i.e., the one with the largest Euclidean distance between its components) direct outside (located between ear and toe), and the longest direct inside.

For the feline data set (99,732 triangles) shown in Figure 9, we detected 12,398 couple points. Figure 9a shows all detected direct inside couple points, Figure 9b does so for all direct outside couple points. Note that we filtered out couple points which are very close to each other for visualization. Figure 9c shows the three longest direct outside couple points, the largest direct inside couple point is shown in Figure 9d.

The "Freezing Old Woman" data set shown in Figure 10 consists of 9,995 triangles. Figure 10b shows the most important direct inside couple points, while Figure 10c shows the longest direct inside couple point.

5.1 Computing the Maximal/Minimal Distance of Surfaces

Given two surfaces, the computation of the minimal distance between them is a relevant problem in computer graphics and can be used for instance for collision detection or 3D path planning. This problem is far more complicated than the computation of the shortest distance between a point and a surface. If the surfaces are given as simple triangular meshes (i.e., without considering interpolated normals), the shortest distance may appear between two vertices, between a vertex and an inner point of a triangle, or between inner points of two edges. In the following, we consider piecewise linear surfaces with piecewise linear normals, a surface model often used in computer graphics as a compromise between simplicity and efficiency on the one side and (often visual) smoothness

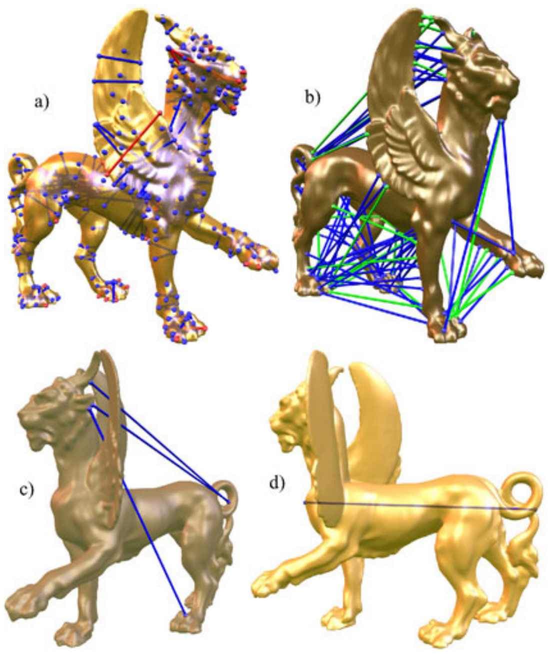


Fig. 9. Couple points of feline: a) all direct inside, b) direct outside, c) longest direct outside, d) longest direct inside

on the other side. For them, the solution generally appears at inner points of two triangles.

Theorem 2 states that the shortest distance between two surfaces is given by a couple point. Hence, to get the shortest distance between two surfaces $\mathbf{x}_1$ and $\mathbf{x}_2$, we can use $C(\mathbf{x}_1, \mathbf{x}_2)$ and select the couple point with the shortest Euclidean distance of its components. In a similar way we can compute the largest distance between two surfaces. Considering a single surface, we can also compute the shortest and largest distance as the distance of the components of couple points which are completely inside or outside the surface.

Figure 11a shows all detected direct outside couple points between the camel and the feline model for a certain relative position between them. (Note that in this picture we filtered out couple points which are very close to each other in both components.) Among them, the shortest couple point gives the shortest distance between the surfaces, as shown Figure 1 (middle). Figures 11b and 10c show the detected largest distance for the feline and the Freezing Old Woman data set, respectively.

Using our algorithm to compute the shortest distances between surfaces, the computing time is essentially the time necessary to extract all couple points (see Section 5). We are not aware of timings of any pre-existing solutions when the considered meshes contain normals at the vertices.

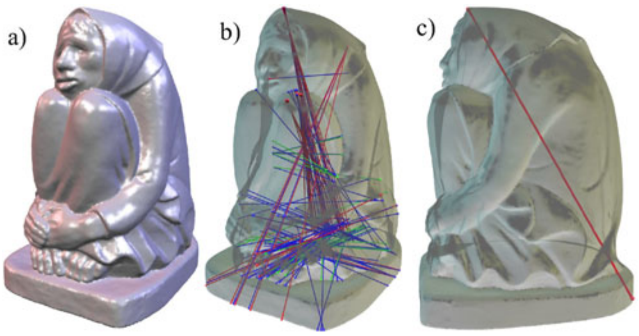


Fig. 10. a) Freezing Old Woman, b) most important direct inside couple points, c) longest direct inside couple point

5.2 Approximations of the Shortest Geodesic Paths between Two Points

Given two points $\mathbf{x}_1$, $\mathbf{x}_2$ on a surface, the computation of the shortest geodesic path connecting them is a rather expensive process which requests a global analysis of the surface. Here, couple points provide a way of computing a fast approximation of the shortest geodesic path. The basic idea is to apply a backward integration of $\mathbf{v}^d$ starting from the double point $(\mathbf{x}_1, \mathbf{x}_2)$. The motivation behind this approach is that $\mathbf{v}^d$ points into the direction of steepest ascent of the Euclidean distance of the components of a double point. Hence, a backward integration of $\mathbf{v}^d$ can be considered as a Greedy algorithm to obtain the shortest path between $\mathbf{x}_1$ and $\mathbf{x}_2$: at every integration step the integrator tries to reduce the Euclidean distance as much as possible.

Doing a backward integration of $\mathbf{v}^d$ starting from $(\mathbf{x}_1, \mathbf{x}_2)$, two cases are possible

- The components of the double points collapse to a single point $(\mathbf{x}_e, \mathbf{x}_e)$. Figure 12a gives an illustration.
- The integration gets stuck in a couple point $({\mathbf{x}_1}^c, {\mathbf{x}_2}^c)$. Figure 12b illustrates this.

In the first case the algorithm stops, and the shortest path is the union of the two components of the integrated double curve. In the second case we need an algorithm to "get out" of the couple point, i.e., we need a shortest path between ${\mathbf{x}_1}^c$ and ${\mathbf{x}_2}^c$. Then the path between $\mathbf{x}_1$ and $\mathbf{x}_2$ is the union of the components of the backward integration from $(\mathbf{x}_1, \mathbf{x}_2)$ to $({\mathbf{x}_1}^c, {\mathbf{x}_2}^c)$ and the shortest path between ${\mathbf{x}_1}^c$ and ${\mathbf{x}_2}^c$. Figure 12c illustrates this.

In order to get the shortest path between the components of a couple point $({\mathbf{x}_1}^c, {\mathbf{x}_2}^c)$, we apply a pre-process to get the shortest paths between the components of all detected couple points. We integrate the separating double curves from each couple point as introduced in Section 3. This way, 8 double stream lines are emanating from a couple point. Among them, we consider all that collapse into a single point. The shortest path of them is the solution for $({\mathbf{x}_1}^c, {\mathbf{x}_2}^c)$.

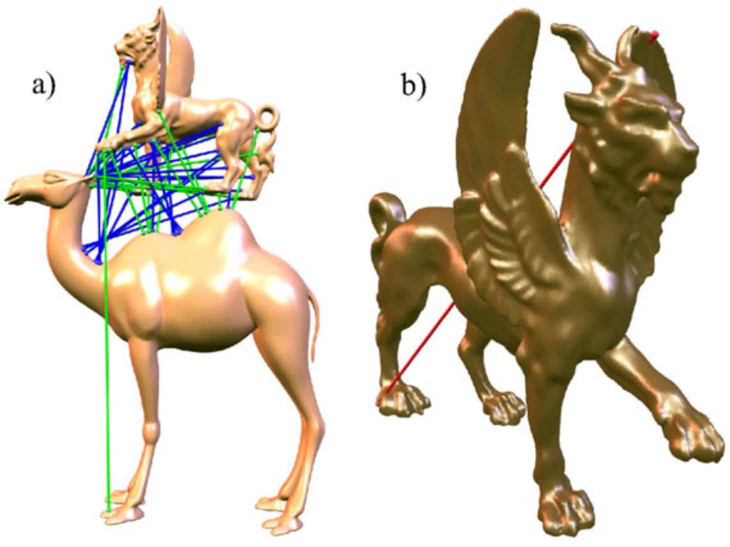


Fig. 11. a) all direct outside couple points between feline and camel, b) longest distance of feline

Figure 13a illustrates an example where 6 of the 8 double stream lines collapse to a single point, while the remaining two get stuck in couple points. In case that all 8 separation double lines get stuck in couple points $(\mathbf{x}_{1i}{}^c, \mathbf{x}_{2i}{}^c)$, we recursively compute the shortest path as the union of the components of the integration double curve from $(\mathbf{x}_1{}^c, \mathbf{x}_2{}^c)$ to $(\mathbf{x}_{1i}{}^c, \mathbf{x}_{2i}{}^c)$ and the shortest path between $\mathbf{x}_{1i}{}^c$ and $\mathbf{x}_{2i}{}^c$. This algorithm is recursively carried out in a breadth first manner until a path for all couple points is found. Figure 13b gives an illustration.

The resulting data structure of our pre-processing is a distance-weighted sparse graph where each node represents a couple point. From each node, 8 edges are leaving which represent the integrated double separation lines. These edges end either in the same node (in case that the integrated double line collapses into a point) or in another node (representing the couple point in which the integration gets stuck). If each edge is assigned with the path length of the corresponding double stream line, the problem of finding a shortest path between $(\mathbf{x}_1{}^c$ and $(\mathbf{x}_2{}^c$ is equivalent to finding a short loop in the graph. This way, we found solutions for all couple points for very few iteration steps.

Once the pre-process for a surface is done, the algorithm to detect a shortest path for a pair of surface points has just the cost of a numerical double stream line integration. Obviously, the cost of this depends on step size and accuracy of the chosen integration technique. Nevertheless it has a linear worst case complexity and is easily possible at interactive frame rates on standard personal computers.

Figure 14 shows two examples of computing the approximation of the shortest path between two surface points (yellow). In both examples, the backward integration of $\mathbf{v}_d$ collapsed into a single point without touching a couple point.

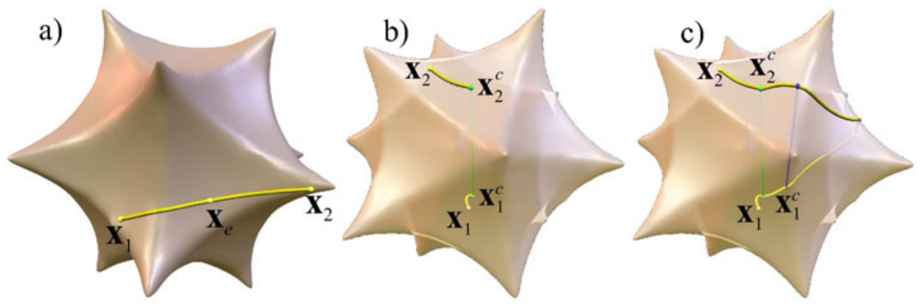


Fig. 12. a) Backward integration of $\mathbf{v}^d$ starting from $(\mathbf{x}_1, \mathbf{x}_2)$: the components of the double points collapse to a single point $\mathbf{x}_e$. b) Integration of $\mathbf{v}^d$ gets stuck in a couple point $({\mathbf{x}_1}^c, {\mathbf{x}_2}^c)$. c) The shortest path is completed by adding the shortest path between ${\mathbf{x}_1}^c$ and ${\mathbf{x}_2}^c$.

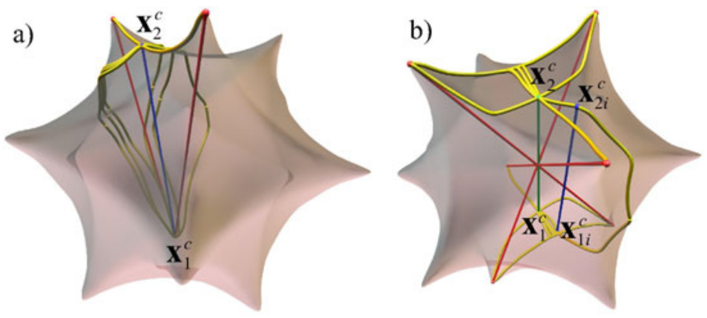


Fig. 13. a) The 8 separation double lines starting from a couple point $({\mathbf{x}_1}^c, {\mathbf{x}_2}^c)$ (blue): 6 collapse to single points. b) All separation lines from $({\mathbf{x}_1}^c, {\mathbf{x}_2}^c)$ (green) get stuck in a couple point: the shortest path is computed as the union of the components of the integrated double line from $({\mathbf{x}_1}^c, {\mathbf{x}_2}^c)$ to $({\mathbf{x}_{1i}}^c, {\mathbf{x}_{2i}}^c)$ and the shortest path between ${\mathbf{x}_{1i}}^c$ and ${\mathbf{x}_{2i}}^c$.

Figure 1 (left) shows an example where the found shortest path is passing through 5 couple points.

Generally, our approach to get a shortest path between two surface points shares the advantages and disadvantages of all Greedy algorithms. It is fast, but is always possible to construct extreme examples in which the algorithm produces solutions far way from the globally optimal one. Figure 15 compares examples of our solutions (yellow lines) with the perfect geodesics (white lines). Figure 15a shows the coincidence between the lines while a certain difference can be observed in Figure 15b. However, the advantage of our approach is that it computes a path between two surface points only by a univariate numerical integration where estimators of real geodesics between two points have a higher complexity.

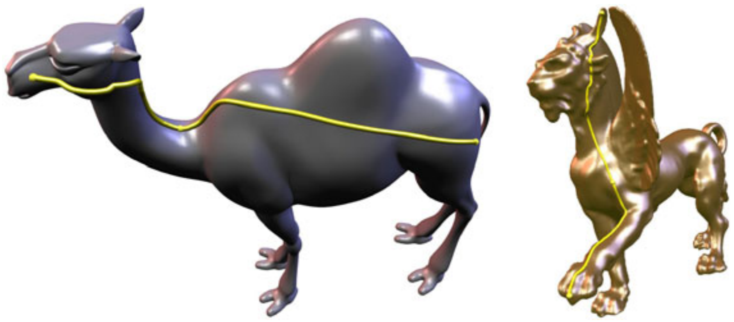


Fig. 14. Fast approximation of the shortest path between two surface points

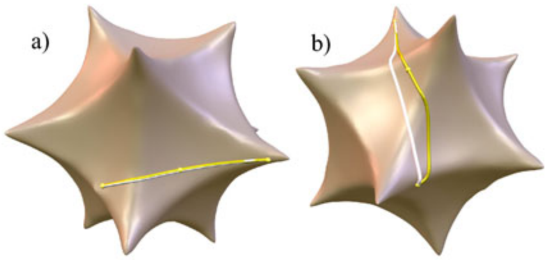


Fig. 15. Comparison between real geodesics (white lines) and our solution (yellow lines)

5.3 Computing Stabilizing Connectors between Parts of a Surface

Quite a number of classical statues and sculptures have lost parts because they got broken (see Figure 16a for an example). To prevent these accidents for

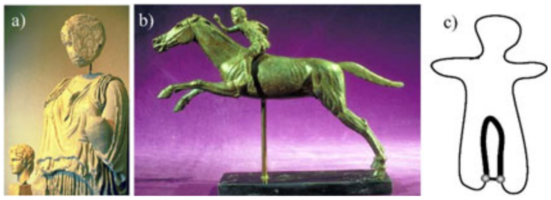


Fig. 16. a) A (real) statue with broken parts. b) A stabilizing connector for a real statue. c) A candidate couple point to be a stabilizing connector: the Euclidean distance between the points is rather short in comparison to the shortest path on the surface.

instance during a transportation, stabilizing connectors can be included. These are static sticks which are connected to certain parts of the surface to prevent the breaking of parts. Figure 16b illustrates an example. Although the stability of a surface is a well-studied feature [6,16], couple points provide a heuristic approach to find optimal stabilizing connectors. An "optimal" stabilizing connector should be a compromise between size, visual appearance and impact. It should be as small as possible in order to not disturb the visual impression of the sculpture, but it should stabilize the sculpture as much as possible. We believe that direct outside couple points are good candidates for stabilizing connectors because they touch the surface parallel to the normals and hence have a maximal effect of the stabilizing forces onto the surface.

Our approach to find the optimal couple points to be used as stabilizing connectors is to consider the ratio between $\|\mathbf{x_2}^c - \mathbf{x_1}^c\|$ and the length of the shortest path between $\mathbf{x_1}^c$ and $\mathbf{x_2}^c$ for all outside direct couple points $(\mathbf{x_2}^c, \mathbf{x_1}^c)$. The smaller this ratio, the better the couple point is suited to be a stabilizing connector. A small value of $\|\mathbf{x_2}^c - \mathbf{x_1}^c\|$ ensures a small disturbance in the visual impression, while a relatively long path between $\mathbf{x_1}^c$ and $\mathbf{x_2}^c$ gives a good impact of the connector. As an example, consider the components of an outside direct couple point $(\mathbf{x_2}^c, \mathbf{x_1}^c)$ to be located on the two legs of a human statue where the position of the legs is rather parallel. Then the Euclidean distance between $\mathbf{x_1}^c$ and $\mathbf{x_2}^c$ is rather small in comparison to the shortest path between $\mathbf{x_1}^c$ and $\mathbf{x_2}^c$, hence the couple point is a good candidate for being a stabilizing connector. Figure 16c gives an illustration.

Figure 17 shows the 7 best suited couple points to be stabilizing connectors

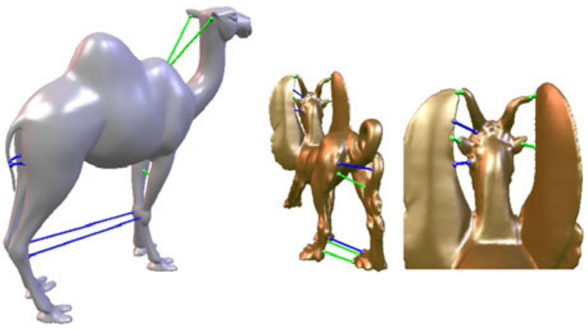


Fig. 17. Best couple points to serve as stabilizing connectors

for the camel data set and the best 11 connectors for the feline data set, a front view is shown in Figure 1 (right).

6 Conclusions

In this paper we have made the following contributions: We introduced the concept of couple points as a new global feature on surfaces. We applied local feature extracting techniques, namely a Morse theoretic approach on a 4D scalar field, to extract and classify couple points. We proposed a recursive algorithm to extract couple points for triangular meshes where the vertices are equipped with a normal. We applied couple points to compute the minimal and maximal distance between surfaces. We applied couple points to compute a fast approximation of the shortest geodesic path between two surface points. Finally, we proposed stabilizing connectors of a surface as appropriate couple points.

Nevertheless there is a number of open problems concerning couple points which are subject to future research. An interesting direction would be persistence of couple points under surface perturbation or surface deformation. For instance, we do not yet have control over the changes of couple points when the surfaces are continuously moved or deformed. In this case, couple points may drift on the surfaces, and they may appear and disappear at certain events. A careful study of these effects may make couple points applicable also to dynamic surfaces.

References

1. Alliez, P., Cohen-Steiner, D., Devillers, O., Lévy, B., Desbrun, M.: Anisotropic polygonal remeshing. ACM Transactions on Graphics 22(3), 485–493 (2003)
2. Chen, J., Han, Y.: Shortest paths on a polyhedron. In: Symposium on Computational Geometry, pp. 360–369 (1990)
3. Culver, T., Keyser, J., Manocha, D.: Accurate computation of the medial axis of a polyhedron. In: Proc. ACM Symp. Solid Model. Appl., pp. 179–190 (1999)
4. Edelsbrunner, H., Harer, J., Natarajan, V., Pascucci, V.: Morse-smale complexes for piecewise linear 3-manifolds. In: Proc. 19th Sympos. Comput. Geom. 2003, pp. 361–370 (2003)
5. Edelsbrunner, H., Harer, J., Zomorodian, A.: Hierarchical morse complexes for piecewise linear 2-manifolds. In: Proc. 17th Sympos. Comput. Geom. 2001 (2001)
6. Efimov, N.: Flächenverbiegungen im Grossen. Akademie-Verlag, Berlin (1957) (in German)
7. Ferrand, E.: On the Bennequin Invariant and the Geometry of Wave Fronts. Geometriae Dedicata 65, 219–245 (1997)
8. Goldfeather, J., Interrante, V.: A novel cubic-order algorithm for approximating principal directions vectors. ACM Transactions on Graphics 23(1), 45–63 (2004)
9. Guéziec, A.: Meshsweeper: Dynamic point-to-polygonal-mesh and applications. IEEE Transactions on Visualization and Computer Graphics 7(1), 47–61 (2001)
10. Hahmann, S., Bonneau, G.P.: Smooth polylines on polygon meshes. In: Brunnett, G., Hamann, B., Müller, H. (eds.) Geometric Modeling for Scientific Visualization, pp. 69–84. Springer, Heidelberg (2003)
11. Hahmann, S., Belyaev, A., Buse, L., Elber, G., Mourrain, B., Rössl, C.: Shape interrogation. In: de Floriani, L., Spagnuolo, M. (eds.) Shape Analysis and Structuring. ch. 1, Mathematics and Visualization, pp. 1–52. Springer, Berlin (2008)
12. Hilaga, M., Shinagawa, Y., Kohmura, T., Kunii, T.: Topology matching for fully automatic similarity estimation of 3D shapes. In: Proc. SIGGRAPH, pp. 203–212 (2001)
13. Hofer, M., Pottmann, H.: Energy-minimizing splines in manifolds. In: Proc. SIGGRAPH, pp. 284–293 (2004)
14. Kuiper, N.H.: Double normals of convex bodies. Israel J. Math. 2, 71–80 (1964)
15. Kimmel, R., Sethian, J.A.: Computing geodesic paths on manifolds. Proc. Natl. Acad. Sci. USA 95(15), 8431–8435 (1998)
16. Minagawa, T., Rado, R.: On the infinitesimal rigidity of surfaces. Osaka Math. J. 4, 241–285 (1952)

17. Mitchell, J.: Geometric shortest paths and network optimization. In: Sack, J.R., Urrutia, J. (eds.) Handbook of Computational Geometry, pp. 633–701. Elsevier, Amsterdam (1998)
18. Ni, X., Garland, M., Hart, J.: Fair morse functions for extracting the topological structure of a surface mesh. In: Proc. SIGGRAPH, pp. 613–622 (2004)
19. Petitjean, S.: A survey of methods for recovering quadrics in triangle meshes. ACM Computing Surveys 34(2) (2001)
20. Pham-Trong, V., Biard, L., Szafran, N.: Pseudo-geodesics on three-dimensional surfaces and pseudo-geodesic meshes. Numerical Algorithms 26(4), 305–315 (2001)
21. Polthier, K., Schmies, M.: Straightest geodesics on polyhedral surfaces. In: Hege, H.C., Polthier, K. (eds.) Mathematical Visualization, pp. 135–150. Springer, Heidelberg (1998)
22. Rusinkiewicz, S.: Estimating curvatures and their derivatives on triangle meshes. In: 3DPVT, pp. 486–493 (2004)
23. Sheeny, D., Armstrong, C., Robinson, D.: Shape description by medial axis construction. IEEE Transactions on Visualization and Computer Graphics 2, 62–72 (1996)
24. Surazhsky, V., Surazhsky, T., Kirsanov, D., Gortler, S.J., Hoppe, H.: Fast exact and approximate geodesics on meshes. ACM Transactions on Graphics 24(3), 553–560 (2005)
25. Theisel, H., Rössl, C., Zayer, R., Seidel, H.P.: Normal based estimation of the curvature tensor for triangular meshes. In: Proc. Pacific Graphics, pp. 288–297 (2004)
26. Xin, S.Q., Wang, G.J.: Improving Chen and Han's algorithm on the discrete geodesic problem. ACM Transactions on Graphics 28(4), 104:1–104:8 (2009)

Chordal Cubic Spline Quasi Interpolation

P. Sablonnière, D. Sbibih, and M. Tahrichi⋆

Paul.Sablonniere@insa-rennes.fr,
sbibih@yahoo.fr, mtahrichi@hotmail.com

Abstract. This paper studies cubic spline quasi-interpolation of parametric curves through sequences of points in any space dimension. We show that if the parameter values are chosen by chord length, the order of accuracy is four. We also use this chordal cubic spline quasi interpolant to approximate the arc length derivatives and the length of the parametric curve.

Keywords: Chordal parametrization, Spline quasi interpolant.

1 Introduction

Fitting a smooth curve through a sequence of points $\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_n$, in $\mathbb{R}^d$, with $d \geq 2$, is a basic problem in geometric modelling and computer graphics. The usual approach is, firstly, choosing parameter values $t_0 < t_1 < \cdots < t_n$, and secondly finding a parametric curve $\mathbf{c} : [t_0, t_n] \to \mathbb{R}^d$ such that

$$\mathbf{c}(t_i) = \mathbf{p_i}, \quad i = 0, 1, \ldots, n.$$

Popular examples of $\mathbf{c}$ are polynomial curves of degree at most n for n small, and spline curves for larger n.

This method was proposed as early as 1967 by Ahlberg, Nilson, and Walsh [1]. But recently, M. S. Floater has investigated more deeply the problem and has explained what has often been observed empirically. Indeed, in [5] he proves that the chord length parameter values yield full approximation order when $\mathbf{c}$ is a polynomial curve of degree at most three, and he develops an algorithm for obtaining full approximation order for arbitrary degree. This study is continued in [6] in the case of cubic spline interpolation, it is proved that if the parameter values are chosen by chord length, the order of accuracy is four. Moreover, it is shown that the derivatives and the length of the original curve are approximated respectively by the arc-length derivatives and the length of the cubic spline interpolant.

Assume that the points $\mathbf{p}_i$ are samples, $\mathbf{p}_i = \mathbf{f}(s_i)$, from a parametric curve $\mathbf{f} : [a, b] \to \mathbb{R}^d$, $d \geq 2$, with $a \leq s_0 < \cdots < s_n \leq b$ and $\mathbf{f}$ is supposed to be parameterized with respect to arc length, i.e., $\mathbf{f}$ is a continuously differentiable function such that $|\mathbf{f}'(s)| = 1$, for all $s \in [a, b]$, where $|.|$ is the Euclidean norm in $\mathbb{R}^d$. Then instead of using cubic spline interpolation as in [6], we propose here to

⋆ Research supported by AI MA/08/182.

choose the parametric curve $\mathbf{c}$ as a C^2 cubic spline quasi interpolant (abbr. QI) based on discrete values of f. The interest of such an operator is that it needs neither the solution of a system of linear equations nor derivative values.

More explicitly, we are interested in choosing parameter values

$$t_0 < t_1 < \cdots < t_n,$$

such that a cubic spline quasi interpolant $Q_\mathbf{p}$ given by

$$Q_\mathbf{p}\mathbf{f}(x) = \sum_{i=1}^{n+3} \mu_{i,\mathbf{p}}(\mathbf{f}) B_i(x),$$

where $\mu_{i,\mathbf{p}}(\mathbf{f})$ are linear combinations of the values $\mathbf{p}_i = \mathbf{f}(s_i)$, and B_i are the cubic B-splines on the interval $[t_0, t_n]$ endowed with the partition $\{t_i,\ i = 0, \ldots, n\}$, has full approximation order, i.e., for all $\mathbf{f} \in C^4([a, b])$

$$d_P(\mathbf{f}, Q_\mathbf{p}\mathbf{f}) = O(\mathbf{h}_s^4) \text{ as } h_s \to 0. \tag{1}$$

Here, $\mathbf{h}_s := \max_{0 \le i \le n-1}(s_{i+1} - s_i)$, and d_p is the metric

$$d_P(\mathbf{f}, Q_\mathbf{p}\mathbf{f}) = \inf_\phi ||\mathbf{f} o \phi - Q_\mathbf{p}\mathbf{f}||,$$

where $||.||$ is the maximum norm, and the infimum is taken over strictly increasing C^1 functions $\phi : [t_0, t_n] \to [s_0, s_n]$ with $\phi(t_0) = s_0$ and $\phi(t_n) = s_n$. It is shown in [8] that, for a pair of regular curves f_1 and f_2, there holds $d_P(f_1, f_2) \ge d_H(f_1, f_2)$ where d_H is the Hausdorff metric.

The main result of the paper is to show that under the chord length parameterization

$$t_{i+1} - t_i = |\mathbf{f}(s_{i+1}) - \mathbf{f}(s_i)|, \quad i = 0, 1, \ldots, n-1,$$

the full approximation order (1) is guaranteed for all $\mathbf{f} \in C^4[a, b]$, i.e., cubic spline quasi interpolant have fourth order accuracy. Later we show that the length of $\mathbf{f}$ is approximated by the length of $Q_\mathbf{p}\mathbf{f}$ with fourth order accuracy. We also show that the derivatives of $\mathbf{f}$ are approximated by the arc-length derivatives of $Q_\mathbf{p}\mathbf{f}$ to the same accuracy as for functions. Finally, we give some numerical examples illustrating these results.

2 Cubic Quasi-Interpolant

In this section we recall some definitions and properties of a cubic spline quasi-interpolant based on discrete function values, see e.g. [2,3,7,10]. Let $X = \{x_0, x_1, \ldots, x_n\}$ be a partition of a bounded interval $I = [x_0, x_n]$. For $1 \le i \le n$, let $h_i = x_i - x_{i-1}$ be the length of the subinterval $I_i = [x_{i-1}, x_i]$. Let $S_3(X)$ be the $n+3$-dimensional space of cubic splines of class C^2 on this partition.

A basis of this space is formed by B-splines $\{B_j,\ j \in J\}$, $J = \{1, 2, \ldots, n+3\}$. With these notations, $\text{supp}(B_j) = [x_{j-4}, x_j]$, and $\mathcal{N}_j = \{x_{j-3}, \ldots, x_{j-1}\}$ is the set of the three interior knots in the support of B_j . As usual, we add multiple knots at the endpoints: $a = x_0 = \cdots = x_{-3}$ and $b = x_n = \cdots = x_{n+3}$.

The C^2 cubic spline quasi interpolant used here is the spline operator given by :

$$Q_3 g = \sum_J \mu_j(g) B_j, \tag{2}$$

where the coefficient functionals $\mu_j(g)$ are determined by solving a linear system of equations given by the exactness of Q_3 on $\mathbb{P}_3$, the space of cubic polynomials functions, see e.g. [2,7,9,10], for more details. More precisely, setting $g_i = g(x_i)$, $i = 0 \dots, n$, we have

$$\begin{aligned}
\mu_1(g) &= g_0, \quad \mu_{n+3}(g) = g_n,\\
\mu_2(g) &= -\frac{2h_2(h_2+h_3)+h_1(2h_2+h_3)}{3(h_1+h_2)(h_1+h_2+h_3)} g_0 + \frac{(h_1+h_2)(h_1+h_2+h_3)}{3h_2(h_2+h_3)} g_1\\
&\quad - \frac{h_1^2(h_1+h_2+h_3)}{3h_2(h_1+h_2)h_3} g_2 + \frac{h_1^2(h_1+h_2)}{3h_3(h_2+h_3)(h_1+h_2+h_3)} g_3,\\
\mu_{n+2}(g) &= -\frac{2h_{n-1}(h_{n-1}+h_{n-2})+h_n(2h_{n-1}+h_{n-2})}{3(h_n+h_{n-1})(h_n+h_{n-1}+h_{n-2})} g_n\\
&\quad + \frac{(h_n+h_{n-1})(h_n+h_{n-1}+h_{n-2})}{3h_{n-1}(h_{n-1}+h_{n-2})} g_{n-1} - \frac{h_n^2(h_n+h_{n-1}+h_{n-2})}{3h_{n-1}(h_n+h_{n-1})h_{n-2}} g_{n-2}\\
&\quad + \frac{h_n^2(h_n+h_{n-1})}{3h_{n-2}(h_{n-1}+h_{n-2})(h_n+h_{n-1}+h_{n-2})} g_{n-3},\\
\mu_j(g) &= -\frac{h_{j-1}^2}{3h_{j-2}(h_{j-2}+h_{j-1})} g_{j-3} + \frac{(h_{j-2}+h_{j-1})^2}{3h_{j-2}h_{j-1}} g_{j-2}\\
&\quad - \frac{h_{j-2}^2}{3h_{j-1}(h_{j-2}+h_{j-1})} g_{j-1}, \quad \text{for } 3 \le j \le n+1.
\end{aligned} \tag{3}$$

Actually, it is well known (see e.g. [4], chapter 5) that there exists constants $C_k,\ k = 0, 1, 2, 3$, such that, for any function $g \in C^4(I)$,

$$||g^{(k)} - Q_3 g^{(k)}||_I \le C_k h^{4-k} ||g^{(4-k)}||_I, \quad k = 0, 1, 2, 3, \tag{4}$$

where $h = \max_{1 \le j \le n} h_j$ and $||g||_I = \max_{x \in I} |g(x)|$.

3 Chord Length Parameterization

We consider the chord length parameterization

$$\begin{aligned}
t_{i+1} - t_i &= |\mathbf{p_{i+1}} - \mathbf{p_i}|\\
&= |\mathbf{f}(s_{i+1}) - \mathbf{f}(s_i)|, \quad i = 0, \dots, n-1.
\end{aligned} \tag{5}$$

It was established in [5] that if $f \in C^2[a, b]$ then for $i = 0, \dots, n-1$,

$$|\varDelta t_i - \varDelta s_i| = O((\varDelta s_i)^3), \text{ as } h_s \to 0. \tag{6}$$

This implies that for $\mathbf{h}_s$ small enough, $\Delta s_i/2 \le \Delta t_i \le \Delta s_i$, and therefore $\mathbf{h}_s/2 \le \mathbf{h}_t \le \mathbf{h}_s$, where $\mathbf{h}_t = \max_{1\le j\le n}(t_j - t_{j-1})$.

Let $Q_\mathbf{p}$ be the operator defined by

$$Q_\mathbf{p}\mathbf{f} = \sum_J \mu_{j,\mathbf{p}}(\mathbf{f})B_j, \tag{7}$$

where $\{B_i,\ i \in J\}$ is the basis of the cubic spline space $S_3(T)$ on the interval $[t_0, t_n]$ endowed with the partition $T = \{t_i,\ i = 0,\ldots,n\}$, and $\mu_{j,\mathbf{p}}$ are the coefficient functionals given in (3) but using the values $\mathbf{p}_i = \mathbf{f}(s_i)$.

Theorem 1. *Suppose that $Q_\mathbf{p}$ is given by (7) based on the chord length parameterization (5). Further, suppose that $\mathbf{h}_s/\min_{0\le i\le n-1}\Delta s_i$ is bounded. If $\mathbf{f} \in C^4[a,b]$ then*

$$d_P(\mathbf{f}, Q_\mathbf{p}\mathbf{f}) = O(\mathbf{h}_s^4) \text{ as } \mathbf{h}_s \to 0. \tag{8}$$

The main idea of the proof is to study the reparameterization

$$\mathbf{g}(t) = \mathbf{f}(\phi(t)), \quad t \in [t_0, t_n],$$

where $\phi : [t_0, t_n] \to \mathbb{R}$ is given by the C^2 cubic spline function satisfying

$$\begin{aligned} \phi(t_i) &= s_i, \quad i = 0,\ldots,n, \\ \phi'(t_i) &= 1, \quad i = 0, n. \end{aligned} \tag{9}$$

We begin by giving some bounds on the derivatives of ϕ. We denote by C a generic constant independent of $\mathbf{h}_t$.

Lemma 1. *Let ϕ be the function given by (9) and $\{t_i,\ i = 0,\ldots,n\}$ are given by (5). Then, for $i = 0, 1, \ldots, n-1$*

$$||\phi' - 1||_{[t_i,t_{i+1}]} \le C\mathbf{h}_t^2;\ ||\phi''||_{[t_i,t_{i+1}]} \le C\mathbf{h}_t;\ ||\phi'''||_{[t_i,t_{i+1}]} \le C\frac{\mathbf{h}_t}{\Delta t_i}. \tag{10}$$

Proof. The proof is given in [4].

Proof (of Theorem 1). The equation of ϕ' in (10) shows that ϕ is increasing when $\mathbf{h}_s$ is small enough, which guarantees that $\mathbf{g}$ is well-defined. On the other hand we have

$$Q_\mathbf{p}\mathbf{f} = \sum_J \mu_{j,\mathbf{p}}(\mathbf{f})B_j,$$

where $\mu_{j,\mathbf{p}}(\mathbf{f})$ are the coefficient functionals in (3) based on values $\mathbf{f}(s_i)$. From (9) we deduce that they use the values of $\mathbf{g}$ at $\{t_i,\ i = 0,\ldots,n\}$. Thus

$$Q_\mathbf{p}\mathbf{f} \equiv Q_3\mathbf{g},$$

and by taking $k = 0$ in (4), we deduce that for $i = 1,\ldots,n-1$,

$$||\mathbf{g} - Q_\mathbf{p}\mathbf{f}||_{[t_0,t_n]} \le C_0\mathbf{h}_t^4||\mathbf{g}^{(4)}||_{[t_0,t_n]},$$

which clearly establishes (8) as long as $\mathbf{g}^{(4)}$ is bounded as $\mathbf{h}_s \to 0$ (we recall that $\mathbf{h}_t \leq \mathbf{h}_s$). To show this we use the formula of Faà di Bruno,

$$\mathbf{g}^{(j)}(t) = \sum_{k_1+\cdots+jk_j=j} \frac{j!}{k_1!..k_j!}\mathbf{f}^{(k_1+\cdots+k_j)}(\phi(t))\left(\frac{\phi'(t)}{1!}\right)^{k_1}\cdots\left(\frac{\phi^{(j)}(t)}{j!}\right)^{k_j}, \quad (11)$$

and since $\mathbf{f}'$, $\mathbf{f}''$, $\dots$, $\mathbf{f}^{(4)}$ are bounded by assumption, it suffices that the derivatives $\phi', \phi'', \dots, \phi^{(4)}$ are bounded, which is a direct consequence of Lemma 1 and the fact that $\mathbf{h}_s / \min_{0\leq i\leq n-1} \Delta s_i$ is bounded. This completes the proof. □

4 Estimating Curve Length

In this section we approximate the length of $\mathbf{f}$ by the length of $Q_p\mathbf{f}$. We notice that the length of $\mathbf{f}$ is the same as the length of $\mathbf{g}$ and so it is sufficient to compare the lengths of $Q_p\mathbf{f}$ and $\mathbf{g}$. Henceforth, we will denote by $\boldsymbol{\sigma}$ the function $Q_p\mathbf{f}$. In the following theorem we show that we get the same order accuracy as for functions, namely $O(\mathbf{h}_s^4)$.

Theorem 2. *If* $\mathbf{f} \in C^4[a,b]$, *then*

$$L(\boldsymbol{\sigma}) - L(\mathbf{f}) = O(\mathbf{h}_s^4), \quad \mathbf{h}_s \to 0. \quad (12)$$

Proof. From inequality (4),

$$||e||_{[t_0,t_n]} = O(\mathbf{h}_s^4), \quad ||e'||_{[t_0,t_n]} = O(\mathbf{h}_s^3), \text{ as } \mathbf{h}_s \to 0, \quad (13)$$

where $\mathbf{e}(t) = \mathbf{g}(t) - \boldsymbol{\sigma}(t)$. It is easy to see that

$$|\boldsymbol{\sigma}'| - |\mathbf{g}'| = -2\frac{\mathbf{e}'\mathbf{g}'}{|\boldsymbol{\sigma}'| + |\mathbf{g}'|} + \frac{\mathbf{e}'.\mathbf{e}'}{|\boldsymbol{\sigma}'| + |\mathbf{g}'|},$$

then by using $|\mathbf{g}'| = \phi'$ and equation (13) we deduce that $|\mathbf{g}'|$ and $|\boldsymbol{\sigma}'|$ are bounded away from zero as $\mathbf{h}_s \to 0$, thus

$$\begin{aligned} L(\boldsymbol{\sigma}) - L(\mathbf{f}) &= \int_{t_0}^{t_n} |\boldsymbol{\sigma}'(t)| - |\mathbf{g}'(t)| dt \\ &= -2\int_{t_0}^{t_n} \frac{\mathbf{e}'(t)\mathbf{g}'(t)}{|\boldsymbol{\sigma}'(t)| + |\mathbf{g}'(t)|} dt + O(\mathbf{h}_s^6). \end{aligned}$$

On the other hand, the quasi interpolant $Q_\mathbf{p}$ is an interpolant at the end points, which gives $\mathbf{e}(t_0) = \mathbf{e}(t_n) = 0$, then an integration by parts implies

$$-\int_{t_0}^{t_n} \frac{\mathbf{e}'(t)\mathbf{g}'(t)}{|\boldsymbol{\sigma}'(t)| + |\mathbf{g}'(t)|} dt = \int_{t_0}^{t_n} \mathbf{e}(t)\frac{\partial}{\partial t}\left(\frac{\mathbf{g}'(t)}{|\boldsymbol{\sigma}'(t)| + |\mathbf{g}'(t)|}\right) dt.$$

The right-hand side in this equality is the error of the quadrature formula based on the cubic quasi interpolant $Q_\mathbf{p}$, with weight function $w(t) = \frac{\partial}{\partial t}\left(\frac{\mathbf{g}'(t)}{|\boldsymbol{\sigma}'(t)|+|\mathbf{g}'(t)|}\right)$. With the help of (4) and (11), that this error is in $O(\mathbf{h}_t^4)$ and also in $O(\mathbf{h}_s^4)$, which proves (12). □

5 Estimating Arc Length Derivatives

From (4), we show that for $\mathbf{f} \in C^4[a,b]$ we have

$$||\mathbf{g}^{(k)} - \boldsymbol{\sigma}^{(k)}|| = O(\mathbf{h}^{4-k}), \quad k = 0, \ldots, 3. \tag{14}$$

Thus the derivatives of $\boldsymbol{\sigma}$ approximate the derivatives of $\mathbf{g}$. Furthermore, we observe that the derivatives of $\mathbf{f}$, being arc length, are the same as the arc length derivatives of $\mathbf{g}$. Thus it is enough to look at the error between the arc length derivatives of $\boldsymbol{\sigma}$ and those of $\mathbf{g}$. Let us denote the arc length derivatives by

$$\dot{\mathbf{g}}(t) = \frac{\mathbf{g}'(t)}{|\mathbf{g}'(t)|}, \quad \dot{\boldsymbol{\sigma}}(t) = \frac{\boldsymbol{\sigma}'(t)}{|\boldsymbol{\sigma}'(t)|}.$$

It is easy to see that

$$\dot{\mathbf{g}} - \dot{\boldsymbol{\sigma}} = \frac{\mathbf{g}' - \boldsymbol{\sigma}'}{|\mathbf{g}'|} + \frac{\boldsymbol{\sigma}'}{|\boldsymbol{\sigma}'||\mathbf{g}'|}\left(|\boldsymbol{\sigma}'| - |\mathbf{g}'|\right).$$

Since $|\mathbf{g}'|$ and $|\boldsymbol{\sigma}'|$ are bounded as $\mathbf{h}_s \to 0$, equation (14) implies that

$$||f' \circ \phi - \dot{\boldsymbol{\sigma}}||_{[t_0,t_n]} = ||\dot{\mathbf{g}} - \dot{\boldsymbol{\sigma}}|| = O(\mathbf{h}_s^3).$$

Similarly, for the second and the third derivatives, using (14) we find that

$$||f'' \circ \phi - \ddot{\boldsymbol{\sigma}}||_{[t_0,t_n]} = ||\ddot{\mathbf{g}} - \ddot{\boldsymbol{\sigma}}|| = O(\mathbf{h}_s^2),$$

$$||f''' \circ \phi - \dddot{\boldsymbol{\sigma}}||_{[t_0,t_n]} = ||\dddot{\mathbf{g}} - \dddot{\boldsymbol{\sigma}}|| = O(\mathbf{h}_s).$$

Specializing to the interpolation points gives estimates that are independent of ϕ.

Theorem 3. *if* $\mathbf{f} \in C^4[a,b]$ *we have*

$$f'(s_i) - \dot{\boldsymbol{\sigma}}(t_i) = O(\mathbf{h}_s^3),\ f''(s_i) - \ddot{\boldsymbol{\sigma}}(t_i) = O(\mathbf{h}_s^2),\ f'''(s_i) - \dddot{\boldsymbol{\sigma}}(t_i) = O(\mathbf{h}_s).$$

Note that the second equation implies that the curvature of $\ddot{\boldsymbol{\sigma}}$ at t_i approximates the curvature of $\mathbf{f}$ at s_i to order $O(\mathbf{h}_s^2)$.

6 Numerical Results

We have implemented algorithms for the construction of the chordal cubic QI $Q_{\mathbf{p}}$ approximating smooth curves through a sequence of points in $\mathbb{R}^d$, $d = 2, 3$.

Example 1: We consider data-points in $\mathbb{R}^2$ sampled from a curve with arc-length parametric representation:

$$\mathbf{f}(s) = \sqrt{\frac{2}{17}}\left(\frac{s}{\sqrt{2}} + 1\right)\left(\cos\left(4\log\left[\frac{s}{\sqrt{2}} + 1\right]\right), \sin\left(4\log\left[\frac{s}{\sqrt{2}} + 1\right]\right)\right),$$

Table 1. Errors and orders of $\boldsymbol{\sigma}$

k	$d_p(\mathbf{f},\boldsymbol{\sigma})$	Order	$\lvert L(\boldsymbol{\sigma}) - L(\mathbf{f})\rvert$	Order
1	6.58E-02		1.77E-01	
2	5.47E-03	3.59	2.05E-02	3.12
3	5.29E-04	3.37	1.58E-03	3.70
4	4.51E-05	3.56	1.07E-04	3.88
5	3.61E-06	3.64	6.97E-06	3.95

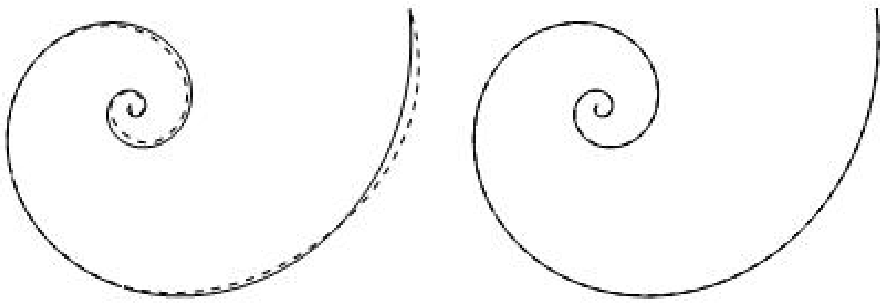


Fig. 1. Chordal C^2 cubic QI for $k = 1$, (left), and chordal C^2 cubic QI for $k = 2$, (right)

at the values $\mathbf{s}^k = \{s_i^k, i = 0, \ldots, 2^{k+3}\}$ for each $k = 0, \ldots, 5$, where the points $\mathbf{s}^0 = \{-1.3, -1.1, -0.8, 0, 1.2, 1.5, 2.1, 3.1, 6\}$ are chosen deliberately to be non-uniform, and for $k \geq 1$, s_i^k for $i = 0, \ldots, 2^{k+3}$ are given by

$$\begin{cases} s_i^k &= s_m^{k-1}, \quad \text{for } i = 2m \\ s_i^k &= \frac{s_m^{k-1}+s_{m+1}^{k-1}}{2}, \quad \text{for } i = 2m+1. \end{cases} \tag{15}$$

For each $k \geq 1$, the QI $Q_{\mathbf{p}}$ was computed using the chord length parameterization. The distance between $\boldsymbol{\sigma}$ and $\mathbf{f}$ was computed numerically by taking the maximum distance form $\boldsymbol{\sigma}$ to $\mathbf{f}$ over 100 uniformly sampled points, and the obtained results are shown in Column 2 of Table 1. Further, we give in Column 4 of the same table, the error between the length of $\boldsymbol{\sigma}$ and the length of $\mathbf{f}$. Numerical approximated orders are given in columns 3 and 5.

Figure 1 shows the curves $\boldsymbol{\sigma}$ based on chordal parameter values for $k = 1, 2$, with the original curve $\mathbf{f}$. In this figure the continuous curve is the original and the dashed one is its approximation.

Example 2: We consider data-points in $\mathbb{R}^3$ sampled from a curve with arc-length parametric representation :

$$\mathbf{f}(s) = \frac{5}{\sqrt{26}}\left(\cos(x), \sin(x), \frac{x}{5}\right),$$

Table 2. Errors and orders of $\boldsymbol{\sigma}$

k	$d_p(\mathbf{f}, \boldsymbol{\sigma})$	order	$\lvert L(\boldsymbol{\sigma}) - L(\mathbf{f})\rvert$	order
1	2.11E-01		–	
2	1.94E-02	3.45	1.01E-01	–
3	1.32E-03	3.87	7.21E-03	3.80
4	8.47E-05	3.96	4.74E-04	3.92
5	5.31E-06	3.99	2.87E-05	4.04

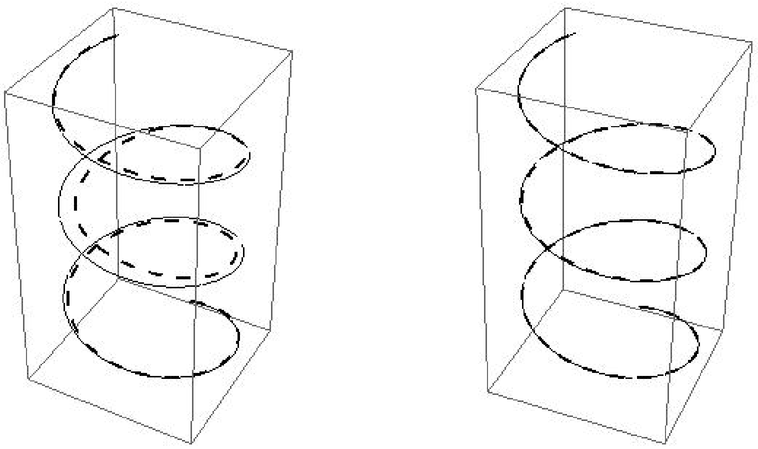


Fig. 2. Chordal C^2 cubic QI for $k = 1$, (left), and chordal C^2 cubic QI for $k = 2$, (right)

at the values $\mathbf{s}^k = \{s_i^k, i = 0, \ldots, 2^{k+3}\}$ for each $k = 0, \ldots, 5$, where $\mathbf{s}^0 = \{0, 2.1, 3.3, 5.7, 8.4, 12.1, 13.5, 16.3, 18\}$ are chosen deliberately to be non-uniform, and for $k \geq 1$, s_i^k for $i = 0, \ldots, 2^{k+3}$ are chosen as in (15). Column 2 in Table 2 shows the distance between $\boldsymbol{\sigma}$ and $\mathbf{f}$, for different values of k, computed numerically as in Example 1, while in Column 4 of this table we give the error $\lvert L(\boldsymbol{\sigma}) - L(\mathbf{f})\rvert$. We list numerical approximated orders in columns 3 and 5. Figure 2 shows the curves $\boldsymbol{\sigma}$ (dashed curve) based on chordal parameter values for $k = 1, 2$, with the original curve $\mathbf{f}$ (continuous curve).

From the above examples and other numerical experiments, the numerical approximated orders agree with the theoretical ones. Further we see, as in Figures 1,2, that the original curve is well approximated by the spline curve $\boldsymbol{\sigma}$.

7 Conclusion

By using chord length parameterization we have obtained full approximation order for C^2 cubic spline quasi interpolation, and we have given estimations for

length and arc length derivatives for curves in $\mathbb{R}^d$, $d \geq 2$. We have illustrated the theory with numerical examples and we think that the results obtained here are comparable to those obtained by using cubic spline interpolation.
One perspective of this work is, by using the same ideas given by Floater for the interpolants, to find a parameterization that guaranties the full approximation order for spline quasi interpolation of degree m, $m > 3$.

References

1. Ahlberg, J.H., Nilson, E.N., Walsh, J.L.: The theory of splines and their applications. Academic Press, New York (1967)
2. Barrera, D., Ibañez, M.J., Sbibih, D., Sablonnière, P.: Near best univariate discrete quasi-interpolants on non-uniform partitions. Constr. Approx. 28, 237–251 (2008)
3. Chen, G., Chui, C.K., Lai, M.J.: Construction of real-time spline quasi-interpolation schemes. Approx. Theory Appl. 4, 61–75 (1988)
4. DeVore, R.A., Lorentz, G.G.: Constructive approximation. Springer, Berlin (1993)
5. Floater, M.S.: Arc length estimation and the convergence of parametric polynomial interpolation. BIT 45, 679–694 (2005)
6. Floater, M.S.: Chordal cubic spline interpolation is fourth order accurate. IMA J. Numer. Anal. 26(1), 25–33 (2006)
7. Lee, B.G., Lyche, T., Schumaker, L.L.: Some examples of quasi-interpolants constructed from local spline projectors. In: Lyche, T., Schumaker, L.L. (eds.) Mathematical Methods for Curves and Surfaces, Oslo 2000, pp. 243–252 (2000)
8. Lyche, T., Mørken, K.: A metric for parametric approximation. In: Laurent, P.J., Le Méhauté, A., Schumaker, L.L. (eds.) Curves and Surfaces, pp. 311–318. A. K. Peters, Wellesley (1994)
9. Rabut, C.: Higher level m-harmonic cardinal B-splines. Numer. Algorithms 2, 63–84 (1992)
10. Sablonnière, P.: Univariate spline quasi-interpolants and applications to numerical analysis. Rend. Sem. Mat. Univ. Pol. Torino 63(2), 107–118 (2005)

Multiple Subdivision Schemes

Tomas Sauer

Lehrstuhl für Numerische Mathematik,
Justus–Liebig–Universität Gießen,
Heinrich–Buff–Ring 44,
D–35392 Gießen, Germany
Tomas.Sauer@math.uni-giessen.de

Abstract. Motivated by the concept of directionally adapted subdivision for the definition of shearlet multiresolution, the paper considers a generalized class of multivariate stationary subdivision schemes, where in each iteration step a scheme and a dilation matrix can be chosen from a given finite set. The standard questions of convergence and refinability will be answered as well as the continuous dependence of the resulting limit functions from the selection process. In addition, the concept of a *canonical factor* for multivariate subdivision schemes is introduced, which follows in a straightforward fashion from algebraic properties of the scaling matrix and takes the role of a smoothing factor for symbols.

1 Introduction

Stationary subdivision is a well–studied subject ever since and even quite before the fundamental monograph [1]. In its most general form, a (scalar) stationary subdivision scheme associates to a multiinfinite sequence $c : \mathbb{Z}^s \to \mathbb{R}$ another sequence

$$c' = S_a c = \sum_{\alpha \in \mathbb{Z}^s} a\,(\cdot - \Xi\alpha)\; c(\alpha), \tag{1}$$

where the *mask* $a : \mathbb{Z}^s \to \mathbb{R}$ is a *finitely supported* sequence and Ξ is a so–called *expanding matrix*, i.e., a matrix all whose eigenvalues are larger than 1 in modulus. The sequence c' is then interpreted as a function on the finer grid $\Xi^{-1}\mathbb{Z}^s$ and by iteration of the subdivision operator one obtains a process that eventually approaches a limit function defined on the continuum $\mathbb{R}^s$ (or not – convergence of subdivision schemes is always an issue).

In this paper, we consider a slightly more general case of stationary subdivision where at any level of iteration there is a finite set of subdivision schemes, i.e., a finite family of masks and scaling matrices to choose from. Nevertheless, any individual step is stationary in the sense of (1): whatever happens at a certain position of c' depends only on the behavior in a certain *neighborhood* of c, hence *what* happens is independent of *where* it happens. Of course, the subdivision process and in particular the resulting limit now depend on which choice of subdivision scheme is made in each step.

The reason for such a construction is not a generalization for the sake of generalization, it occured naturally in the context of *shearlets*, cf. [8], where two subdivision schemes where introduced that were basedon scaling matrices which used the same scaling component but a different shear content. Based on properties of such a multiple subdivision scheme, a variation of multiresolution analysis can be constructed that allows for a filter bank treatment of discrete shearlets.

The paper first treats the basic aspects of multiple subdivision, like convergence and the associated concept of refinability. In Section 4, we take a slight distraction from the path and consider *canonical factors* which allow for a particularly simple construction of convergent schemes and can be considered a natural generalization of the B–splines in the univariate case. What is more interesting, however, is the fact that these objects follow quite straightforward from purely algebraic considerations of the masks in terms of classical matrix factorizations.

2 Multiple Subdivision and Refinability

We consider subdivision with limit functions in $L_p(\mathbb{R}^s)$ based on sequences from $\ell_p(\mathbb{Z}^s)$, both equipped with the standard norms for $1 \le p < \infty$,

$$\|f\|_p = \|f\|_{L_p(\mathbb{R}^s)} = \left(\int_{\mathbb{R}^s} |f(x)|^p \, dx \right)^{1/p},$$

$$\|c\|_p = \|c\|_{\ell_p(\mathbb{Z}^s)} = \left(\sum_{\alpha \in \mathbb{Z}^s} |c(\alpha)|^p \right)^{1/p}.$$

For $p = \infty$, we replace, as usually, $L_\infty(\mathbb{R}^s)$ by $C_u(\mathbb{R}^s)$, the space of uniformly continuous functions, and use the standard ∞–norms

$$\|f\|_\infty = \sup_{x \in \mathbb{R}^s} |f(x)|, \qquad \|c\|_\infty = \sup_{\alpha \in \mathbb{Z}^s} |f(\alpha)|.$$

Instead of introducing a different symbol for these spaces, "$L_\infty(\mathbb{R}^s)$" will always have to be understood as $C_u(\mathbb{R}^s)$ in the context of this paper. In addition, $\ell_{00}(\mathbb{Z}^s)$ will stand for the finitely supported sequences. Finally, the "semidiscrete" convolution between a function and a sequence is defined like usually as

$$f * c = \sum_{\alpha \in \mathbb{Z}^s} f(\cdot - \alpha) \, c(\alpha).$$

To extend the concept of stationary subdivision to the interaction of multiple subdivision schemes, we fix $m \in \mathbb{N}$ and consider *m masks* $a_j \in \ell_{00}(\mathbb{Z}^s)$, and *scaling matrices* $\Xi_j \in \mathbb{Z}^{s \times s}$, $j \in \mathbb{Z}_m := \{0, 1, \ldots, m-1\}$. As usual, masks are supposed to have finite support and, for simplicity, we choose N large enough such that the support of all masks a_j is contained in $\Omega := [-N, N]^s$, i.e.,

$$a_j(\mathbb{Z}^s \setminus \Omega) = \{0\}, \qquad j \in \mathbb{Z}_m. \tag{2}$$

Scaling matrices are supposed to be expanding ones, hence, all eigenvalues of any Ξ_j are strictly larger than 1 in modulus Another way to express this property is to demand that

$$\lim_{k\to\infty} \left\| \Xi_j^{-k} \right\| = 0, \qquad j \in \mathbb{Z}_m,$$

which means that the *spectral radius* of each inverse Ξ_j^{-1} is less than one.

The basic subdivision operators S_j are still the usual ones,

$$S_j c := \sum_{\alpha\in\mathbb{Z}^s} a_j \left(\cdot - \Xi_j \alpha\right) c(\alpha), \qquad j \in \mathbb{Z}_m, \tag{3}$$

but the subdivision scheme now consists of arbitrary interactions of these operators, so that we have

$$S_\eta := S_{\eta_r} \cdots S_{\eta_1}, \qquad \eta = (\eta_1, \ldots, \eta_r) \in \mathbb{Z}_m^r, \quad r \in \mathbb{N}. \tag{4}$$

The result of such an operator is related to the grid $\Xi_\eta^{-1}\mathbb{Z}^s$, where

$$\Xi_\eta = \Xi_{\eta_r} \cdots \Xi_{\eta_1}, \qquad \eta \in \mathbb{Z}_m^r,$$

is again a scaling matrix if $\left\|\Xi_\eta^{-1}\right\| \to 0$ for $r \to \infty$, independently of η. This in turn is the case if the *joint spectral radius* of the family $\Xi = (\Xi_j \ : \ j \in \mathbb{Z}_m)$ of matrices,

$$\rho\left(\Xi^{-1}\right) = \limsup_{r\to\infty} \max_{\eta\in\mathbb{Z}_m^r} \left\|\Xi_\eta^{-1}\right\|^{1/r},$$

satisfies $\rho\left(\Xi^{-1}\right) < 1$. Any family of matrices that satisfies this condition will be called *jointly expanding* and in the sequel we will always tacitly assume that "our" Ξ is jointly expanding. For convenient abbreviation, we also define the sets

$$H = \bigcup_{r\in\mathbb{N}} \mathbb{Z}_m^r, \qquad H_\infty = \mathbb{Z}_m^{\mathbb{N}} := \{\eta = (\eta_j \ : \ j \in \mathbb{N}) \ : \ \eta_j \in \mathbb{Z}_m\}$$

of finite and infinite index sequences with values in $\mathbb{Z}_m$. The *projection operators* $P_r \ : \ H_\infty \to \mathbb{Z}_m^r$ compute the canonical projection by means of truncation, i.e.,

$$P_r \eta = (\eta_1, \ldots, \eta_r), \qquad \eta \in H_\infty.$$

For a matrix $\Theta \in \mathbb{Z}^{s\times s}$ we define the *average sequences*

$$\mu_p(f, \Theta) := \left(|\det \Theta| \int_{\Theta^{-1}(\alpha+[0,1]^s)} f(t)\, dt \ : \ \alpha \in \mathbb{Z}^s \right), \qquad 1 \le p < \infty \tag{5}$$

and

$$\mu_\infty(f, \Theta) := \left(f\left(\Theta^{-1}\alpha\right) \ : \ \alpha \in \mathbb{Z}^s \right), \tag{6}$$

respectively. There are various equivalent ways to define the convergence of stationary subdivision schemes in $\mathbb{R}^s$, see [3,6], but the concept can be extended naturally by saying that the multiple subdivision scheme based on

$a := (a_j \ : \ j \in \mathbb{Z}_m)$ and $\Xi := (\Xi_j \ : \ j \in \mathbb{Z}_m)$ is *p–convergent* if for any $\eta \in H_\infty$ there exists a *basic limit function* f_η such that

$$\lim_{r \to \infty} \left(\det \Xi_{P_r \eta}\right)^{-1/p} \left\| \mu_p \left(f_\eta, \Xi_{P_r \eta}\right) - S_{P_r \eta} \delta \right\|_{\ell_p(\mathbb{Z}^s)} = 0. \tag{7}$$

An alternative but of course equivalent definition of convergence uses the concept of *test functions*, cf. [3]. A function $\varphi \in L_p(\mathbb{R}^s)$ is called a test function, if it is a compactly supported stable partition of unity, that is, if $\varphi * 1 = \sum_\alpha \varphi(\cdot - \alpha) = 1$ and there exist constants $0 < A \le B \le \infty$ such that $A \|c\|_{\ell_p} \le \|\varphi * c\|_{L_p} \le B \|c\|_{\ell_p}$. Then convergence is also described by

$$\lim_{r \to \infty} \left\| f_\eta - \left(\varphi * S_{P_r \eta} \delta\right) \left(\Xi_{P_r \eta} \cdot\right) \right\| = 0 \tag{8}$$

for either one or any test function. Recall that it has been shown in [3] that if (8) holds for a particular test function, then it holds for any test function.

Since the subdivision schemes are all linear, (7) and (8) are equivalent to the convergence of $S_{P_r \eta} c$ to a limit $f_{\eta,c}$ for *any* initial sequence $c \in \ell_p(\mathbb{Z}^s)$ with

$$f_{\eta,c} = f_\eta * c = \sum_{\alpha \in \mathbb{Z}^s} f_\eta (\cdot - \alpha) \ c(\alpha).$$

Choosing $\eta = (j, j, \ldots) \in H_\infty$, $j \in \mathbb{Z}_m$, we immediately get the following obvious observation, cf. [3,14].

Lemma 1. *If a and Ξ define a convergent subdivision scheme, then so does each of the pairs a_j, Ξ_j, $j \in \mathbb{Z}_m$. Consequently, $S_j 1 = 1$ or, equivalently,*

$$\sum_{\alpha \in \mathbb{Z}^s} a_j \left(\Xi_j \alpha + \xi\right) = 1, \qquad \xi \in X_j := \mathbb{Z}^s / \Xi_j \mathbb{Z}^s, \qquad j \in \mathbb{Z}_m. \tag{9}$$

In addition, we also have that $f_\eta * 1 = 1$, $\eta \in H_\infty$.

Moreover, a simple inductive argument yields for $\eta \in \mathbb{Z}_m^r$ that

$$S_\eta c = \sum_{\alpha \in \mathbb{Z}^s} a_\eta \left(\cdot - \Xi_\eta \alpha\right) \ c(\alpha), \qquad a_\eta = S_{\eta_r} a_{P_{r-1} \eta}. \tag{10}$$

Indeed,

$$\begin{aligned} S_\eta c = S_{\eta_r} S_{P_{r-1}\eta} c &= \sum_{\alpha \in \mathbb{Z}^s} a_{\eta_r} \left(\cdot - \Xi_{\eta_r} \alpha\right) S_{P_{r-1}\eta} c(\alpha) \\ &= \sum_{\alpha \in \mathbb{Z}^s} a_{\eta_r} \left(\cdot - \Xi_{\eta_r} \alpha\right) \sum_{\beta \in \mathbb{Z}^s} a_{P_{r-1}\eta} \left(\alpha - \Xi_{P_{r-1}\eta} \beta\right) \ c(\beta) \\ &= \sum_{\beta \in \mathbb{Z}^s} c(\beta) \sum_{\alpha \in \mathbb{Z}^s} a_{\eta_r} \left(\cdot - \Xi_\eta \beta - \Xi_{\eta_r} \alpha\right) a_{P_{r-1}\eta}(\alpha) \\ &= \sum_{\beta \in \mathbb{Z}^s} \left(S_{\eta_r} a_{P_{r-1}}\right) \left(\cdot - \Xi_\eta \beta\right) \ c(\beta) \\ &= \sum_{\alpha \in \mathbb{Z}^s} a_\eta \left(\cdot - \Xi_\eta \alpha\right) \ c(\alpha) =: a_\eta *_\eta c, \end{aligned}$$

where

$$a_\eta *_\eta c := a_\eta * \uparrow_{\Xi_\eta} c, \qquad \uparrow_\Xi c(\alpha) = \begin{cases} c(\beta), & \alpha = \Xi\beta, \\ 0, & \text{otherwise}, \end{cases} \tag{11}$$

denotes the *upsampled convolution* with upsampling matrix Ξ_η.

All a_η, $\eta \in H$, are of finite support as well and their support is contained in Ω_η where

$$\Omega_\eta := \Omega \cup \bigcup_{s=1}^{r} \Xi_{\eta_r} \cdots \Xi_{\eta_s} \Omega, \qquad \eta = (\eta_1, \dots, \eta_r)\,. \tag{12}$$

This fact is easily observed by noting that, if $\eta = (\widehat{\eta}, j)$, $\widehat{\eta} \in H$, $j \in \mathbb{Z}_m$, then

$$a_\eta(\alpha) = \sum_{\beta \in \mathbb{Z}^s} a_j \left(\alpha - \Xi_{\widehat{\eta}}\right) a_{\widehat{\eta}}(\beta) = \sum_{\beta \in \Omega_{\widehat{\eta}}} a_j \left(\alpha - \Xi_j \beta\right) a_{\widehat{\eta}}(\beta)$$

is zero if $\alpha - \Xi_j \Omega_{\widehat{\eta}} \cap \Omega = \{0\}$, that is, if $\alpha \notin \Omega + \Xi_j \Omega_{\widehat{\eta}}$, and (12) follows by a simple induction. Since all the matrices in Ξ are strictly expanding, there exists $\rho \in (0,1)$ such that $\left\| \Xi_j^{-1} \right\|_\infty \leq \rho$ and therefore

$$\Xi_\eta^{-1} \Omega_\eta = \Xi_\eta^{-1} \Omega \cup \bigcup_{s=1}^{r} \Xi_{\eta_1} \cdots \Xi_{\eta_{s-1}} \Omega \subseteq \frac{1}{1-\rho} \Omega =: \Omega^*, \qquad \eta = (\eta_1, \dots, \eta_r) \in H, \tag{13}$$

that is, the sets $\Xi_\eta^{-1} \Omega_\eta$ are bounded uniformly with respect to η, where we can also assume that the bounding Ω^* is symmetric: $\Omega^* = -\Omega^*$. As usually, this also implies that all basic limit functions have compact support contained in Ω^*, though, of course, theses supports can be significantly smaller than Ω^*.

Next, we observe that all the basic limit functions of a convergent scheme are refinable, though not necessarily individually.

Theorem 1. *If a and Ξ define a convergent subdivision scheme, then, for any $\eta \in H_\infty$,*

$$f_\eta = f_{\widehat{\eta}} * a_{\eta_1}\left(\Xi_{\eta_1} \cdot\right) = \sum_{\alpha \in \mathbb{Z}^s} a_{\eta_1}(\alpha)\, f_{\widehat{\eta}}\left(\Xi_{\eta_1} \cdot - \alpha\right), \qquad \widehat{\eta} = (\eta_2, \eta_3, \cdots)\,. \tag{14}$$

Proof. The proof is a straightforward extension of the one from [8], but will be given here for the sake of completeness. It makes use of the *transition operators*

$$T_\eta := T_{\eta_1} \cdots T_{\eta_r}, \qquad T_j f := \sum_{\alpha \in \mathbb{Z}^s} a_j(\alpha)\, f\left(\Xi_j \cdot - \alpha\right), \quad j \in \mathbb{Z}_m,$$

which are closely related to the subdivision operators by means of

$$\begin{aligned} T_j f * c &= \sum_{\beta \in \mathbb{Z}^s} \sum_{\alpha \in \mathbb{Z}^s} a_j(\alpha)\, f\left(\Xi_j (\cdot - \beta) - \alpha\right)\, c(\beta) \\ &= \sum_{\alpha \in \mathbb{Z}^s} f\left(\Xi_j \cdot - \alpha\right) \sum_{\beta \in \mathbb{Z}^s} a_j\left(\alpha - \Xi_j \beta\right)\, c(\beta) = \left(f * S_j c\right)\left(\Xi_j \cdot\right) \end{aligned}$$

hence, by iterating this identity, we get that T_η and S_η are essentially adjoints with respect to convolution:

$$T_\eta f * c = (f * S_\eta c)(\Xi_\eta \cdot), \qquad \eta \in H. \tag{15}$$

For $r \in \mathbb{N}$, $\eta = (\eta_1, \widehat{\eta}) \in H_\infty$, and any test function φ we then have

$$\begin{aligned} T_{\eta_1} f_{\widehat{\eta}} &= (f_{\widehat{\eta}} * S_{\eta_1}\delta)(\Xi_{\eta_1}) \\ &= \left(\left(f_{\widehat{\eta}} - \left(\varphi * S_{P_{r-1}\widehat{\eta}}\delta\right)\left(\Xi_{P_{r-1}\widehat{\eta}}\right)\right) * S_{\eta_1}\delta\right)(\Xi_{\eta_1}) + (\varphi * S_{P_r\eta}\delta)(\Xi_{P_r\eta}) \end{aligned}$$

so that

$$\begin{aligned} \|T_{\eta_1} f_{\widehat{\eta}} - f_\eta\|_p \le \|S_{\eta_1}\|_p &\left\| f_{\widehat{\eta}} - \left(\varphi * S_{P_{r-1}\widehat{\eta}}\delta\right)\left(\Xi_{P_{r-1}\widehat{\eta}}\right)\right\|_p \\ &+ \|f_\eta - (\varphi * S_{P_r\eta}\delta)(\Xi_{P_r\eta})\|_p \end{aligned}$$

Since for any $j \in \mathbb{Z}_m$ the operator norm $\|S_j\|_p$ is bounded and since the subdivision scheme is convergent, the right side of this inequality converges to zero for $r \to \infty$ while the left hand side is even independent of r. Hence, $T_{\eta_1} f_{\widehat{\eta}} = f_\eta$ as claimed in (14). □

The refinement equation (14) can be easily iterated into the form

$$f_{(\eta,\theta)} = f_\theta * a_\eta(\Xi_\eta) = \sum_{\alpha \in \mathbb{Z}^s} a_\eta(\alpha) f_\theta(\Xi_\eta \cdot - \alpha), \qquad \eta \in H,\ \theta \in H_\infty, \tag{16}$$

where $a_\eta := S_\eta \delta$ is the mask from (10). In particular, this implies that

$$f_\theta * S_\eta c = f_{(\eta,\theta)} * c\left(\Xi_\eta^{-1} \cdot\right). \tag{17}$$

Next, we consider the convergence of multiple subdivision schemes. To that end, we first note that as an immediate consequence of the definition (8) standard methods, cf. [16,17], give us the following reformulation of Lemma 1.

Lemma 2. *If the subdivision scheme based on Ξ and a converges then*

$$a_j^*(z) \in \left\langle z^{\Xi_j} - 1 \right\rangle : \langle z - 1 \rangle, \qquad j \in \mathbb{Z}_m. \tag{18}$$

Let us recall the algebraic notation used in (18). The *symbol* a^* associated to the sequence $a \in \ell_{00}\left(\mathbb{Z}^2\right)$ is the *Laurent polynomial*

$$a^*(z) = \sum_{\alpha \in \mathbb{Z}^s} a(\alpha)\, z^\alpha, \qquad z \in (\mathbb{C} \setminus \{0\})^s,$$

the *(Laurent) ideals* to be considered are

$$\left\langle z^\Xi - 1 \right\rangle := \left\{ \sum_{j=1}^s f_j(z)\left(z^{\xi_j} - 1\right) \ :\ f_j \in \Lambda \right\}, \qquad \langle z - 1 \rangle = \left\langle z^I - 1 \right\rangle,$$

where Λ denotes the ring of Laurent polynomials. Finally, for (Laurent) ideals $\mathcal{I}, \mathcal{J} \subset \Lambda$, the *quotient ideal* is defined as

$$\mathcal{I} : \mathcal{J} = \{ f \in \Lambda \; : \; f \cdot \mathcal{J} \subseteq \mathcal{I} \},$$

cf. [2]. After that little bit of well–known classical concepts, it is worthwhile to remember that for $s = 1$ and $\Xi = kI$, (18) means that the symbol has a factor of the form $1 + x + \cdots + x^{k-1}$, so that it is nothing but the natural counterpart of the "zero at π" condition so well known from wavelet theory.

Polynomial reproduction of the subdivision operators, aka "sum rules", on the other hand, is equivalent to polynomial reproduction of the individual subdivision operators which can most conveniently be expressed in terms of the quotient ideals as well. Indeed, all subdivision operators preserve polynomials of (total) degree up to n if and only if

$$a_j^*(z) \in \left\langle z^{\Xi_j} - 1 \right\rangle^{n+1} : \langle z - 1 \rangle^{n+1}. \tag{19}$$

3 Convergence

Like in the usual stationary case, the convergence of multiple subdivision schemes will be characterized by the existence and joint contractivity of so called *difference schemes* which satisfy the interlacing relationship

$$\nabla S_{a_j} = S_{B_j} \nabla, \qquad \nabla c := \begin{bmatrix} c(\cdot - \epsilon_1) - c \\ \vdots \\ c(\cdot - \epsilon_s) - c \end{bmatrix}, \qquad j \in \mathbb{Z}_m. \tag{20}$$

with the backwards difference operator ∇ whose symbol is

$$\nabla^* = \begin{bmatrix} z_1 - 1 \\ \vdots \\ z_s - 1 \end{bmatrix} =: [z - 1].$$

Defining, in the same spirit, the vectors

$$\left[z^\Theta - 1 \right] = \left[z^{\theta_j} - 1 \; : \; j = 1, \dots, s \right], \qquad \Theta = [\theta_1, \dots, \theta_s] \in \mathbb{Z}^{s \times s},$$

with $[z - 1] = \left[z^I - 1 \right]$, the algebraic equivalent of (20) is

$$[z - 1] \, a_j^*(z) = B_j^*(z) \, \left[z^{\Xi_j} - 1 \right], \qquad j \in \mathbb{Z}_m. \tag{21}$$

In addition, it is well–known that (9) implies, for any $j \in \mathbb{Z}_m$, that there exists a Laurent polynomial B_j^* that satisfies (21), or, equivalently, a *finitely supported* mask $B_j \in \ell_{00}(\mathbb{Z}^s)$ such that (20) holds true.

The masks B_j, $j \in \mathbb{Z}_m$, also define a multiple subdivision scheme (B, Ξ), now acting on *vector valued* data. The *normalized p–joint spectral radius* of these subdivision operators is naturally defined as

$$\rho_p(B, \Xi) := \limsup_{r\to\infty} \sup_{\eta\in\mathbb{Z}_m^r} \left((\det \Xi_\eta)^{-1/p} \sup_{c\in\nabla\ell_p(\mathbb{Z}^s)\setminus\{0\}} \frac{\|S_{B,\eta}c\|_p}{\|c\|_p} \right)^{1/r}. \qquad (22)$$

Note that, as usual in the multivariate case, the norm in (22) is only taken as an operator norm over $\nabla\ell_p(\mathbb{Z}^s)$, which is a proper subspace of $\ell^s(\mathbb{R}^s)$ for $s > 1$. Moreover, the normalization term $(\det \Xi_\eta)^{-1/p}$ is now integrated into the definition of ρ_p in (22) since in the case of multiple subdivision it obviously depends on η.

Theorem 2. *The subdivision scheme based on (a, Ξ) converges if and only if there exist difference schemes (B, Ξ) such that $\rho_p(B, \Xi) < 1$.*

We split the proof of Theorem 2 into several parts, starting with the necessary condition.

Proposition 1. *If the subdivision scheme based on (a, Ξ) converges then there exists a difference scheme (B, Ξ) such that $\rho_p(B, \Xi) < 1$.*

Proof. By Lemma 1, convergence implies (9), the sum rule of order 0, hence $a_j^*(z) \in \langle z^{\Xi_j} - 1\rangle : \langle z - 1\rangle$, and thus there exist $s \times s$–matrix valued masks B_j, $j \in \mathbb{Z}_m$, such that

$$\nabla S_{a,j} = S_{B,j}\nabla, \qquad \text{hence,} \qquad \nabla S_{a,\eta} = S_{B,\eta}\nabla, \quad \eta \in H. \qquad (23)$$

Then, for any $c \in \ell_p(\mathbb{Z}^s)$, any $\eta \in H_\infty$ and $r \in \mathbb{N}$ we have that

$$\begin{aligned} \|S_{B,P_r\eta}\nabla c\|_p &= \|\nabla S_{a,P_r\eta}c\|_p \\ &\le \|\nabla(\mu_p(f_\eta * c, \Xi_{P_r\eta}) - S_{P_r\eta}c)\|_p + \|\nabla\mu_p(f_\eta * c, \Xi_{P_r\eta})\|_p \\ &\le 2\|\mu_p(f_\eta * c, \Xi_{P_r\eta}) - S_{P_r\eta}c\|_p + \|\nabla\mu_p(f_\eta * c, \Xi_{P_r\eta})\|_p. \end{aligned}$$

A simple direct computation gives

$$\mu_p(f_\eta * c, \Xi_{P_r\eta}) = \mu_p(f_\eta, \Xi_{P_r\eta}) *_{P_r\eta} c \qquad (24)$$

with the upsampled convolution from (11), and thus we can apply Lemma 3, which will be proved immediately, to find that

$$\begin{aligned} &\|\mu_p(f_\eta * c, \Xi_{P_r\eta}) - S_{P_r\eta}c\|_p \\ &= \|(\mu_p(f_\eta, \Xi_{P_r\eta}) - a_{P_r\eta}) *_{P_r\eta} c\|_p \le C\|\mu_p(f_\eta, \Xi_{P_r\eta}) - S_{P_r\eta}\delta\|_p \|\nabla c\|_p. \end{aligned}$$

Another (componentwise) application of Lemma 3 and the fact that

$$\|\nabla\mu(f, \Xi)\|_p \le (s\det\Xi)^{1/p}\,\omega_p\left(f, \|\Xi^{-1}\|\right)$$

yield

$$\begin{aligned}&\left\|\nabla \mu_p\left(f_\eta * c, \Xi_{P_r \eta}\right)\right\|_p \\ &\quad \leq C\left\|\nabla \mu_p\left(f_\eta, \Xi_{P_r \eta}\right)\right\|_p\|\nabla c\|_p \leq C\left(s \det \Xi_{P_r \eta}\right)^{1/p} \omega_p\left(f_\eta,\left\|\Xi_{P_r \eta}^{-1}\right\|\right)\|\nabla c\|_p,\end{aligned}$$

so that

$$\begin{aligned}&\left(\det \Xi_{P_r \eta}\right)^{-1/p}\left\|S_{B, P_r \eta} \nabla c\right\|_p \\ &\quad \leq\left(\left(\det \Xi_{P_r \eta}\right)^{-1/p}\left\|\mu_p\left(f_\eta, \Xi_{P_r \eta}\right)-S_{P_r \eta} \delta\right\|_p+s^{1/p} \omega_p\left(f_\eta,\left\|\Xi_{P_r \eta}^{-1}\right\|\right)\right)\|\nabla c\|_p\end{aligned}$$

and since the terms in the parentheses on the right hand side tends to zero for $r \to \infty$, the claim follows by the standards arguments, cf. [3,11]. □

Lemma 3. *Suppose that $a \in \ell_{00}\left(\mathbb{Z}^s\right)$ is supported on Ω_η and satisfies $a *_\eta 1 = 0$. Then there exists a constant $C > 0$ such that for any $c \in \ell_p\left(\mathbb{Z}^s\right)$*

$$\left\|a *_\eta c\right\|_p \leq C\|a\|_p\|\nabla c\|_p . \tag{25}$$

Proof. In order to estimate

$$\left\|a *_\eta c\right\|_p^p=\sum_{\alpha \in \mathbb{Z}^s}\left|\sum_{\beta \in \mathbb{Z}^s} a\left(\alpha-\Xi_\eta \beta\right) c(\beta)\right|^p, \tag{26}$$

we first note that the inner sum only runs over $\beta \in \Xi_\eta^{-1}\left(\alpha+\Omega_\eta\right) \subseteq \Xi_\eta^{-1} \alpha+\Omega^*$. Choose any $\beta^* \in \Xi_\eta^{-1} \alpha+\Omega^*$ and write $c(\beta)$ as the telescoping sum

$$\begin{aligned} c(\beta) & =c\left(\beta^*\right)+\sum_{k=1}^s \sum_{j_k=0}^{\beta_k-\beta_k^*-1} c\left(\beta^*+\left(j_k+1\right) \epsilon_k\right)-c\left(\beta^*+j_k \epsilon_k\right) \\ & =c\left(\beta^*\right)+\sum_{k=1}^s \sum_{j_k=0}^{\beta_k-\beta_k^*-1} \epsilon_k^T \nabla c\left(\beta^*+j_k \epsilon_k\right),\end{aligned}$$

where the sums have to be understood in the sense that $\sum_0^k := \sum_k^0$ whenever $k<0$. This can be rewritten in slightly fancier way as

$$c(\beta)=c\left(\beta^*\right)+\sum_{\gamma \in \Omega^*} y_\gamma^T \nabla c(\beta+\gamma), \qquad y_\gamma \in\left\{0, \epsilon_1, \ldots, \epsilon_s\right\} . \tag{27}$$

Now we substitute (27) into (26), take into account the fact that $a *_\eta 1=0$ and perform the usual routine estimates based on Hölder's inequality for $1 \leq p<\infty$ and $q=\left(1-p^{-1}\right)^{-1}$

$$\left\|a *_\eta c\right\|_p^p=\sum_{\alpha \in \mathbb{Z}^s}\left|\sum_{\beta \in \Xi_\eta^{-1} \alpha+\Omega^*} a\left(\alpha-\Xi_\eta \beta\right)\left(c\left(\beta^*\right)+\sum_{\gamma \in \Omega^*} y_\gamma^T \nabla c(\beta+\gamma)\right)\right|^p$$

$$\begin{aligned}
&\leq (\#\Omega^*)^{2q} \sum_{\alpha\in\mathbb{Z}^s} \sum_{\beta\in\Xi_\eta^{-1}\alpha+\Omega^*} \sum_{\gamma\in\Omega^*} |a(\alpha-\Xi_\eta\beta)|^p \left|y_\gamma^T \nabla c(\beta+\gamma)\right|^p \\
&\leq (\#\Omega^*)^{2q} \sum_{\gamma\in\mathbb{Z}^s} \sum_{\beta\in\mathbb{Z}^s} \sum_{\alpha\in\mathbb{Z}^s} |a(\alpha)|^p \|y_\gamma\|_p^q \|\nabla c(\beta+\gamma)\|_p^p \\
&\leq (\#\Omega^*)^{2q+1} \|a\|_p^p \|\nabla c\|_p^p ,
\end{aligned}$$

to obtain (25). The estimate for $p=\infty$ is even simpler, cf. [12]. □

The converse of Proposition 1 can be formulated as follows.

Proposition 2. *If for (a,Ξ) there exists a difference scheme (B,Ξ) such that $\rho_p(B,\Xi)<1$, then (a,Ξ) admits a convergent subdivision scheme.*

Proof. The proof relies on a result to be found in [4,5], where it is shown that for any expanding matrix Θ there exists a uniformly continuous *cardinal* function ψ that is refinable with respect to Θ, that is, there exists a finitely supported mask $g\in\ell_{00}(\mathbb{Z}^s)$ such that $\psi=(\psi*g)(\Theta\cdot)$. Moreover, these functions – which are closely connected B–splines – can be normalized such that $\psi*1=1$. Let ψ_j denote such functions that are refinable with respect to Ξ_j and mask p_j, $j\in\mathbb{Z}_m$.

We fix a test function g and consider, for given $\eta\in H_\infty$, the sequence

$$g^{(r)} := g * S_{a,P_r\eta} c\left(\Xi_{P_r\eta}\cdot\right)$$

of functions. Now, with $j:=\eta_{r+1}$,

$$\begin{aligned}
\left\|g^{(r+1)}-g^{(r)}\right\|_p &\leq \left\|\psi_j * \left(S_{a_j}-S_{p_j}\right) S_{P_r\eta}c\right\|_p \\
&\quad + \left\|(g-\psi_j)*S_{P_r\eta}c\right\|_p + \left\|(g-\psi_j)*S_{P_{r+1}\eta}c\right\|_p .
\end{aligned}$$

As usual, the fact that $\left(S_{a_j}-S_{p_j}\right)1=0$ implies that there exists a matrix mask $H_j\in\ell^{s\times s}(\mathbb{Z}^s)$ such that $S_{a_j}-S_{p_j}=S_{H_j}\nabla$, while [8, Lemma 4.19] also ensures the existence of a uniformly continuous, compactly supported matrix function G such that $(g-\psi_j)*c=G_j*\nabla c$. Hence, with

$$C := \max_{j\in\mathbb{Z}_m} \left(\|\psi_j\|_\infty \left\|S_{H_j}\right\|_p + \left(1+\left\|S_{B_j}\right\|_p\right) \|G_j\|_p \right)$$

we get that

$$\left\|g^{(r+1)}-g^{(r)}\right\|_p \leq C \det \Xi_{P_r\eta}^{-1/p} \left\|S_{B,P_r\eta}\nabla c\right\|_p , \tag{28}$$

where the bound on the right hand side is independent of $j=\eta_{r+1}$ and tends to zero for $r\to\infty$. Hence, $g^{(r)}$ is a Cauchy sequence whose limit function is $f_{\eta,c}=f_\eta*c$. This is nothing but the convergence of the subdivision scheme. □

On H_∞ we define the natural metric by setting

$$[0,1]\ni d(\eta,\eta') := |\eta-\eta'| := \sum_{j=1}^\infty m^{-j} \left|\eta_j-\eta_j'\right| , \qquad \eta,\eta'\in H_\infty . \tag{29}$$

Note that if η and η' are interpreted as m–adic expansions of numbers $x, x' \in [0,1]$, then $|x - x'|$ is small if $d(\eta, \eta')$ is small, but not vice versa. With (29) in hand, we can state that the basic limit functions depend in a "reasonable" way on the index $\eta \in H_\infty$.

Corollary 1. *If (a, Ξ) admits a convergent subdivision scheme then the mapping $\eta \mapsto f_\eta$ is continuous.*

Proof. We have already shown that if (a, Ξ) admits a convergent subdivision scheme, then

$$\|f_\eta - g * S_{P_r \eta} \delta \left(\Xi_{P_r \eta}\right)\| \le C \, \rho^r$$

for some $\rho < 1$ and a constant C independent of η and r. Now, given $\eta \in H_\infty$ and $\varepsilon > 0$, we get for any η' such that $|\eta - \eta'| < m^{-r}$ with $r > \log_\rho \frac{\varepsilon}{2C}$ that $P_r \eta = P_r \eta'$ and hence

$$\|f_\eta - f_{\eta'}\| \le \|f_\eta - g * S_{P_r \eta} \delta \left(\Xi_{P_r \eta}\right)\| + \|f_{\eta'} - g * S_{P_r \eta'} \delta \left(\Xi_{P_r \eta}\right)\| \le 2C \, \rho^r = \varepsilon,$$

so that f_η depends continuously on η. □

Another well–known sufficient condition for convergence is existence and *stability* of the refinable function. Recall that a function $f \in L_p(\mathbb{R}^s)$ is called *stable* if there exist constants $0 < A \le B < \infty$ such that

$$A \, \|c\|_p \le \|f * c\|_p \le B \, \|c\|_p, \qquad c \in \ell_p(\mathbb{Z}^s).$$

Then we have the following result.

Proposition 3. *If for some $1 \le p \le \infty$ the refinement equation (14) has, for any η, a stable solution in $L_p(\mathbb{R}^s)$, then (a, Ξ) is L_p–convergent.*

Proof. Fix η. For $r \in \mathbb{N}$ let $\widehat{\eta} \in H_\infty$ be such that $\eta = (P_r \eta, \widehat{\eta})$. Then the stability of $f_{\widehat{\eta}}$ and (17) with $c = \delta$ yield

$$\begin{aligned}
&\left(\det \Xi_{P_r \eta}\right)^{-1/p} \|\mu_p\left(f_\eta, \Xi_{P_r \eta}\right) - S_{P_r \eta} \delta\|_{\ell_p} \\
&\quad \le A^{-1} \left(\det \Xi_{P_r \eta}\right)^{-1/p} \|f_{\widehat{\eta}} * \mu_p\left(f_\eta, \Xi_{P_r \eta}\right) - f_{\widehat{\eta}} * S_{P_r \eta} \delta\|_{L_p} \\
&\quad = A^{-1} \left(\det \Xi_{P_r \eta}\right)^{-1/p} \left\| f_{\widehat{\eta}} * \mu_p\left(f_\eta, \Xi_{P_r \eta}\right) - f_\eta \left(\Xi_{P_r \eta}^{-1}\right) \delta \right\|_p \\
&\quad = A^{-1} \|f_{\widehat{\eta}} * \mu_p\left(f_\eta, \Xi_{P_r \eta}\right)\left(\Xi_{P_r \eta}\right) - f_\eta\|_p,
\end{aligned}$$

and it has been shown, for example in [3], that the term on the right hand side of this expression converges to zero for $r \to \infty$ since the matrices are jointly expanding. □

4 Canonical Factors and Further Convergence Issues

As mentioned before in Lemma 1, containment of $a_j^*(z)$ in the quotient ideal $\left\langle z^{\Xi_j} - 1 \right\rangle : \langle z - 1 \rangle$ is equivalent to $S_j 1 = 1$, i.e., to the existence of the difference mask and hence a fundamental necessary condition for the convergence of a subdivision scheme. We will consider this issue first for the case $m = 1$, hence Ξ now stands for a one–element family of expanding matrices, that is, a generic expanding matrix.

For given $\Xi \in \mathbb{Z}^{s\times s}$, there is a simple canonical representation of this quotient ideal. To that end, recall the *Smith factorization* of any integer matrix, cf. [9], which is a decomposition $\Xi = \Theta\Sigma\Lambda$ into integer matrices such that Θ, Λ are *unimodular*, i.e. $|\det\Theta| = |\det\Lambda| = 1$, and Σ is diagonal with positive entries.

Proposition 4. *If $\Xi = \Theta\Sigma\Lambda$ is a Smith factorization, then*

$$\left\langle z^{\Xi} - 1\right\rangle : \langle z - 1\rangle = \left\langle z^{\Xi} - 1, \psi_{\Xi}(z)\right\rangle, \tag{30}$$

where

$$\psi_{\Xi}(z) := \prod_{j=1}^{s}\sum_{k=0}^{\sigma_j-1} z^{k\theta_j} = \prod_{j=1}^{s}\frac{z^{\sigma_j\theta_j}-1}{z^{\theta_j}-1}, \tag{31}$$

and where θ_j are the column vectors of Θ and σ_j the diagonal elements of Σ.

The proof of Proposition 4 is based on the following simple observation.

Lemma 4. *If p is any s–vector of (Laurent) polynomials and Θ a unimodular matrix, then $\left\langle p^{\Theta}(z) - 1\right\rangle = \langle p(z) - 1\rangle$.*

Proof. We consider the associated varieties $Z = \{z : p(z) = 1\}$ and $Z_\Theta = \left\{z : p^{\Theta}(z) = 1\right\}$ where in the case of Laurent polynomials we would have to replace Z and Z_Θ by their intersections with $(\mathbb{C}\setminus\{0\})^s$, cf. [13]. For $z \in Z$ we take Θth powers of both sides of $p(z) = 1$ to obtain $1 = 1^{\Theta} = p^{\Theta}(z)$, hence $z \in Z_\Theta$. Conversely, for $z \in Z_\Theta$ we have $1 = p^{\Theta}(z)$ and therefore

$$1 = 1^{\Theta^{-1}} = \left(p^{\Theta}(z)\right)^{\Theta^{-1}} = p^{\Theta\Theta^{-1}}(z) = p(z),$$

so that also $z \in Z$. Hence, the varieties agree and so do the ideals. □

Proof. (of Proposition 4) Using the substitution $y = z^{\Theta}$ we get

$$\left\langle z^{\Xi} - 1\right\rangle : \langle z - 1\rangle = \left\langle z^{\Theta\Sigma\Lambda} - 1\right\rangle : \langle z - 1\rangle = \left\langle y^{\Sigma\Lambda} - 1\right\rangle : \left\langle y^{\Theta^{-1}} - 1\right\rangle$$

and two applications of Lemma 4 yield

$$\left\langle z^{\Xi} - 1\right\rangle : \langle z - 1\rangle = \left\langle y^{\Sigma} - 1\right\rangle : \langle y - 1\rangle. \tag{32}$$

The polynomials $y^{\sigma_j} - 1$, $j = 1,\dots,s$, form a universal Gröbner basis for the ideal $\left\langle y^{\Sigma} - 1\right\rangle$, cf. [15], and since the canonical factor

$$\psi_{\Xi}(y) = \prod_{j=1}^{s}\frac{y_j^{\sigma_j}-1}{y_j-1} = \prod_{j=1}^{s}\sum_{k=0}^{\sigma_j-1} y^k$$

belongs to the quotient ideal $\langle y^\Sigma - 1 \rangle : \langle y - 1 \rangle$ and is reduced with respect to the Gröbner (and H-) basis $\left\{ y^\Sigma - 1 \right\}$, cf. [2], it follows easily that

$$\langle y^\Sigma - 1 \rangle : \langle y - 1 \rangle = \langle y^\Sigma - 1, \psi_\Xi (y) \rangle .$$

Replacing y_j by z^{θ_j}, this yields (31). □

Like in the standard case with scaling matrix $2I$, the presence canonical factor ensures the convergence of the associated subdivision scheme.

Theorem 3. *The subdivision scheme (a, Ξ) with $a_j^* = \psi_{\Xi_j}$ converges in L_1.*

In addition, since the auto-convolution of finitely supported L_1–functions is continuous, we immediately have convergence to (uniformly) continuous functions that can be considered the counterpart of B–splines.

Corollary 2. *The subdivision scheme (a, Ξ) with $a_j^* = \psi_{\Xi_j}^2$ converges to a continuous limit function.*

Proof. (of Theorem 3) We first consider the case of $a^* = \psi_\Xi$ for some fixed Ξ. Since for $j = 1, \dots, s$

$$(z_j - 1) \, \psi_\Xi (z) = \left(z^{\theta_j \sigma_j} - 1 \right) \prod_{k \neq j} \sum_{\ell = 0}^{\sigma_k - 1} z^{\ell \theta_k},$$

we can choose the difference symbol B for $a^* = \psi_\Xi$ diagonal with

$$B_{jj}^* (z) = \prod_{k \neq j} \sum_{\ell = 0}^{\sigma_k - 1} z^{\ell \theta_k} = \sum_{\substack{\alpha \le \sigma - 1 \\ \alpha_j = 0}} z^{\Theta \alpha}.$$

Since Θ is unimodular, hence nonsingular, $\Theta\alpha = \Theta\beta$ can happen if and only if $\alpha = \beta$ and therefore all nonzero coefficients of B_{jj} have value 1, hence $\|S_B\|_1 = 1 < \det \Xi$.

This also implies that $\|S_{B,\eta}\|_1 \le 1$ for all $\eta \in H$ and thus the conditions of Theorem 2 are met which allows us to conclude that (a, Ξ) indeed converges in L_1. □

We close these observations on the canonical factors by mentioning that the exponents in the canonical factors are representers for $\mathbb{Z}^s / \Xi \mathbb{Z}^s$. Since Λ and Λ^{-1} are unimodular, and thus $\mathbb{Z}^s = \Lambda \mathbb{Z}^s = \Lambda^{-1}$ we have that $\mathbb{Z}^s / \Xi \mathbb{Z}^s = \left(\Lambda^{-1} \mathbb{Z}^s \right) / \left(\Xi \Lambda^{-1} \mathbb{Z}^s \right)$. In addition, the set $\Xi \Lambda^{-1} [0,1)^s \cap \mathbb{Z}^s = \Xi \Lambda^{-1} [0,1)^s \cap \Lambda^{-1} \mathbb{Z}^s$ contains as many integer elements as $\Xi [0,1)^s \cap \mathbb{Z}^s$, namely $|\det \Xi|$. Hence, the exponents

$$\sum_{j=1}^s \alpha_j \, \theta_j = \Theta \alpha, \qquad \alpha \le \sigma - 1,$$

form a set X^* of representers for $\Xi\mathbb{Z}^s/\mathbb{Z}^s$. Consequently, the canonical factors can also be written as the simplest form of a mask that satisfies (9) by having only the nonzero coefficients $a(\xi) = 1$, $\xi \in X^*$.

Next, we generalize the already mentioned result by Han [5] that ensures the existence of convergent interpolatory refinable functions for arbitrary scaling matrices to multiple subdivision, returning to general $m \in \mathbb{N}$ again. To that end, we define

$$\widehat{a}_j(\omega) := \frac{1}{\det \Xi_j} a^*\left(e^{-i\omega}\right), \qquad j \in \mathbb{Z}_m, \tag{33}$$

so that the refinement equation (14) can be rewritten in Fourier transformed form as

$$\widehat{f}_\eta(\omega) = \widehat{a}_{\eta_1}\left(\Xi_{\eta_1}^{-T}\right) \widehat{f}_{\widehat{\eta}}\left(\Xi_{\eta_1}^{-T}\omega\right), \qquad \eta = (\eta_1, \widehat{\eta}),$$

or, more generally,

$$\widehat{f}_{(\theta,\eta)}(\omega) = \prod_{j=1}^{\ell(\theta)} \widehat{a}_{\theta_j}\left(\Xi_{P_j\theta}^{-T}\omega\right) \widehat{f}\left(\Xi_\theta^{-T}\omega\right), \qquad \theta \in H,\, \eta \in H_\infty, \tag{34}$$

so that the Fourier transform of the refinable function results as the infinite product

$$\widehat{f}_\eta = \prod_{j=1}^{\infty} \widehat{a}_{\eta_j}\left(\Xi_{P_j\eta}^{-T}\,\cdot\right). \tag{35}$$

The subdivision scheme (a, Ξ) is called *interpolatory* if all the schemes (a_j, Ξ_j) are interpolatory, that is

$$a_j(0) = 1, \quad a_j(\Xi_j\cdot) = 0, \qquad j \in \mathbb{Z}_m. \tag{36}$$

This implies that $S_{a_j}c(\Xi_j\cdot) = c$ and thus

$$S_\eta c(\Xi_\eta\cdot) = c. \tag{37}$$

Theorem 4. *Suppose that the masks (a, Ξ) satisfy the following properties:*

1. $\sum_\alpha a_j(\cdot + \Xi_j\alpha) = 1$, $j \in \mathbb{Z}_m$,
2. $\widehat{a}_j(\omega) \geq 0$, $\omega \in \mathbb{R}^s$,
3. (a, Ξ) *is interpolatory,*
4. $\widehat{a}_j(\omega) = 0$ *if and only if* $\omega \in 2\pi\Xi_j^{-T}\mathbb{Z}^s \setminus 2\pi\mathbb{Z}^s$.

Then there exists, for any $\eta \in H_\infty$, a finitely supported continuous refinable function f_η which is, in addition, cardinal, i.e. $f(\alpha) = \delta_{\alpha,0}$, $\alpha \in \mathbb{Z}^s$.

Recall that [5] proves the existence of masks that satisfy the assumptions of Theorem 4 and even gives an explicit construction for these cardinal functions.

Proof. The proof is a rather straightforward extension of the one from [5] which in turn relies on ideas from [10] and is based on studying the infinite product

([35]). We first note that any ω such that $\widehat{f}_\eta(\omega) = 0$ must satisfy $\widehat{a}_{\eta_j}\left(\Xi_{P_j\eta}^{-T}\omega\right) = 0$ for some j, hence $\Xi_{P_j\eta}^{-T}\omega \in 2\pi\Xi_{\eta_j}^{-T}\mathbb{Z}^s \setminus 2\pi\mathbb{Z}^s$. Since $\Xi_{P_j\eta}^{-T} = \Xi_{\eta_j}^{-T}\Xi_{P_{j-1}\eta}^{-T}$, this is equivalent to $\omega \in 2\pi\mathbb{Z}^s$ and since the normalization yields $\widehat{f}_\eta(0) = 1$, we have $\widehat{f}_\eta(\omega) = 0$ if and only if $\omega \in 2\pi\mathbb{Z}^s \setminus \{0\}$. Now, one proceeds as in [10] and shows that the sequence

$$g_n := \chi_{\Xi_{P_n\eta}[-\pi,\pi)^d} \prod_{j=1}^{n} \widehat{a}_{\eta_j}\left(\Xi_{P_j\eta}^{-T}\cdot\right), \qquad n \in \mathbb{N},$$

satisfies $\widehat{g}_n \geq 0$, converges to the L_1–function $\widehat{f}_\theta$, and the continuous inverse Fourier transform f_θ is cardinal since $\widehat{f}_\theta > 0$ on $[-\pi,\pi]^s$.

The final step to show that the subdivision scheme converges is also a standard argument: Since f_η is a compactly supported continuous cardinal function, it is *stable*, and since therefore all refinable functions are stable, the subdivision scheme has to converge. □

5 Geometric Choice of Scaling Matrices

So far, the theory was general in the choice of scaling matrices and masks, and we have already seen that there exists a variety of convergent subdivision schemes and thus of multiply refinable functions for any choice of such scaling matrices. We will now restrict the choice of scaling matrices to more geometric considerations where we fix Ξ_0 and construct Ξ_j, $j \in \mathbb{Z}_{m-1}+1$ accordingly. The basic idea from the 2D shearlet setting, cf. [7,8], was to express the subdivision schemes based on Ξ_j as the subdivision scheme for Ξ_0 but with a suitable transformation of the data before and afterwards. To that end, note that any *unimodular* matrix Θ defines an invertible *dilation operator* D_Θ on $\mathbb{Z}^s$ by $D_\Theta c = c\,(\Theta\cdot)$. Clearly, $D_\Theta^{-1} = D_{\Theta^{-1}}$. Now we choose a family Θ_j, $j \in \mathbb{Z}_m$, of such unimodular matrices with, for simplicity, $\Theta_0 = I$, and consider the subdivision operators

$$\begin{aligned} S_{a_j,\Xi_j} c = D_{\Theta_j} S_{\widetilde{a}_j,\Xi_0} D_{\Theta_j^{-1}} c &= \sum_{\alpha\in\mathbb{Z}^s} \widetilde{a}_j\left(\Theta_j \cdot - \Xi_0\alpha\right)\, c\left(\Theta_j^{-1}\alpha\right) \\ &= \sum_{\alpha\in\mathbb{Z}^s} \widetilde{a}_j\left(\Theta_j\left(\cdot - \Theta_j^{-1}\Xi_0\Theta_j\right)\alpha\right)\, c(\alpha) \end{aligned}$$

so that $\Xi_j := \Theta_j^{-1}\Xi_0\Theta_j$ and $a_j = D_{\Theta_j}\widetilde{a}_j = \widetilde{a}_j\left(\Theta_j\cdot\right)$, $j \in \mathbb{Z}_m$. For example, in the shearlet case Ξ_0 was an *anisotropic scaling matrix* (with 4 and 2 on the diagonal) and the unimodular transformation was set to be the *shear matrix*

$$\Theta_1 = \begin{bmatrix} 1 & -2 \\ 0 & 1 \end{bmatrix} \qquad \text{so that} \qquad \Xi_1 = \begin{bmatrix} 4 & -4 \\ 0 & 2 \end{bmatrix}.$$

This particular choice allowed for a very useful property, namely that the matrix Ξ_η could be always decomposed into a scaling part and a unimodular transformation part, that is

$$\Xi_\eta = \Xi_0^r \Gamma_\eta' = \Gamma_\eta \Xi_0^r, \qquad \eta \in H_r, \quad r > 0, \tag{38}$$

such that Γ_η and Γ'_η are unimodular matrices. Such a system Ξ of scaling matrices will be called a *uniform scaling system.* For the special case $\eta = \epsilon_j$ we thus get

$$\Theta_j^{-1}\Xi_0\Theta_j = \Gamma_j\Xi_0 = \Xi_0\Gamma'_j,$$

hence

$$\Gamma_j = \Theta_j^{-1}\Xi_0\Theta_j\Xi_0^{-1}, \qquad \Gamma'_j = \Xi_0^{-1}\Theta_j^{-1}\Xi_0\Theta_j, \qquad j \in \mathbb{Z}_m. \tag{39}$$

Example 1. In the shearlet case we have

$$\Gamma_1 = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & -2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{4} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -4 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -2 \\ 0 & 1 \end{bmatrix}$$

and

$$\Gamma'_1 = \begin{bmatrix} \frac{1}{4} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & -2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}.$$

Acknowledgement. I want to thank the referees for careful reading and spotting several typos and the organizers of the Avignon conference for the great time and inspiring atmosphere we had there.

References

1. Cavaretta, A.S., Dahmen, W., Micchelli, C.A.: Stationary Subdivision, Memoirs of the AMS, vol. 93 (453). Amer. Math. Soc., Providence (1991)
2. Cox, D., Little, J., O'Shea, D.: Ideals, Varieties and Algorithms. Undergraduate Texts in Mathematics, 2nd edn. Springer, Heidelberg (1996)
3. Dahmen, W., Micchelli, C.A.: Biorthogonal wavelet expansion. Constr. Approx. 13, 294–328 (1997)
4. Derado, J.: Multivariate refinable interpolating functions. Appl. Comp. Harmonic Anal. 7, 165–183 (1999)
5. Han, B.: Compactly supported tight wavelet frames and orthonormal wavelets of exponential decay with a general dilation matrix. J. Comput. Appl. Math. 155, 43–67 (2003)
6. Jia, R.Q.: Subdivision schemes in l_p spaces. Advances Comput. Math. 3, 309–341 (1995)
7. Kutyniok, G., Sauer, T.: From wavelets to shearlets and back again. In: Schumaker, L.L. (ed.) Approximation Theory XII, San Antonio, TX, pp. 201–209. Nashboro Press, Nashville (2007/2008)
8. Kutyniok, G., Sauer, T.: Adaptive directional subdivision schemes and shearlet multiresolution analysis. SIAM J. Math. Anal. 41, 1436–1471 (2009)
9. Marcus, M., Minc, H.: A Survey of Matrix Theory and Matrix Inequalities. Prindle, Weber & Schmidt (1969); paperback reprint, Dover Publications (1992)
10. Micchelli, C.A.: Interpolatory subdivision schemes and wavelets. J. Approx. Theory 86, 41–71 (1996)
11. Micchelli, C.A., Sauer, T.: Regularity of multiwavelets. Advances Comput. Math. 7(4), 455–545 (1997)
12. Micchelli, C.A., Sauer, T.: On vector subdivision. Math. Z. 229, 621–674 (1998)

13. Möller, H.M., Sauer, T.: Multivariate refinable functions of high approximation order via quotient ideals of Laurent polynomials. Advances Comput. Math. 20, 205–228 (2004)
14. Sauer, T.: Stationary vector subdivision – quotient ideals, differences and approximation power. Rev. R. Acad. Cien. Serie A. Mat. 96, 257–277 (2002)
15. Sauer, T.: Lagrange interpolation on subgrids of tensor product grids. Math. Comp. 73, 181–190 (2004)
16. Sauer, T.: Differentiability of multivariate refinable functions and factorization. Advances Comput. Math. 25, 211–235 (2006)
17. Sauer, T.: Multivariate refinable functions, difference and ideals – a simple tutorial. J. Comput. Appl. Math. 221, 447–459 (2008)

On a Linear Programming Approach to the Discrete Willmore Boundary Value Problem and Generalizations

Thomas Schoenemann[1], Simon Masnou[2], and Daniel Cremers[3]

[1] Center for Mathematical Sciences, Lund University, Sweden
tosch@maths.lth.se
[2] Université de Lyon, Université Lyon 1, CNRS UMR 5208,
Institut Camille Jordan, Villeurbanne, France
masnou@math.univ-lyon1.fr
[3] Department of Computer Science, Technical University of Munich,
Garching, Germany
daniel.cremers@in.tum.de

Abstract. We consider the problem of finding (possibly non connected) discrete surfaces spanning a finite set of discrete boundary curves in the three-dimensional space and minimizing (globally) a discrete energy involving mean curvature. Although we consider a fairly general class of energies, our main focus is on the Willmore energy, i.e. the total squared mean curvature. Most works in the literature have been devoted to the approximation of a surface evolving by the Willmore flow and, in particular, to the approximation of the so-called Willmore surfaces, i.e., the critical points of the Willmore energy. Our purpose is to address the delicate task of approximating *global* minimizers of the energy under boundary constraints. The main contribution of this work is to translate the nonlinear boundary value problem into an integer linear program, using a natural formulation involving pairs of elementary triangles chosen in a pre-specified dictionary and allowing self-intersection. The reason for such strategy is the well-known existence of algorithms that can compute *global minimizers* of a large class of linear optimization problems, however at a significant computational and memory cost. The case of integer linear programming is particularly delicate and usual strategies consist in relaxing the integral constraint $x \in \{0, 1\}$ into $x \in [0, 1]$ which is easier to handle. Our work focuses essentially on the connection between the integer linear program and its relaxation. We prove that:

– One cannot guarantee the total unimodularity of the constraint matrix, which is a sufficient condition for the global solution of the relaxed linear program to be always integral, and therefore to be a solution of the integer program as well;
– Furthermore, there are actually experimental evidences that, in some cases, solving the relaxed problem yields a fractional solution.

These facts indicate that the problem cannot be tackled with classical linear programming solvers, but only with pure integer linear solvers. Nevertheless, due to the very specific structure of the constraint matrix here, we strongly believe that it should be possible in the future to design ad-hoc integer solvers that yield high-definition approximations to solutions of several boundary value problems involving mean curvature, in particular the Willmore boundary value problem.

Keywords: mean curvature, Willmore functional, integer linear programming, relaxation, total unimodularity.

1 Introduction

The Willmore energy of an immersed compact oriented surface $f : \Sigma \to \mathbb{R}^N$ with boundary $\partial\Sigma$ is defined as

$$\mathcal{W}(f) = \int_{\Sigma} |H|^2 dA + \int_{\partial\Sigma} \kappa \, ds$$

where H is the mean curvature vector on Σ, κ the geodesic curvature on $\partial\Sigma$, and dA, ds the induced area and length metrics on Σ, $\partial\Sigma$. The Willmore energy of surfaces with or without boundary plays an important role in geometry, elastic membranes theory, strings theory, and image processing. Among the many concrete optimization problems where the Willmore functional appears, let us mention for instance the modeling of biological membranes, the design of glasses, and the smoothing of meshed surfaces in computer graphics. The Willmore energy is the subject of a long-standing research not only due to its relevance to some physical situations but also due to its fundamental property of being conformal invariant, which makes it an interesting substitute to the area functional in conformal geometry. Critical points of $\mathcal{W}$ with respect to interior variations are called Willmore surfaces. They are solutions of the Euler-Lagrange equation $\delta\mathcal{W} = 0$ whose expression is particularly simple when $N = 3$: $\Delta H + 2H(H^2 - K) = 0$, being K the Gauss curvature. It is known since Blaschke and Thomsen [23] that stereographic projections of compact minimal surfaces in $\mathbb{S}^3 \subset \mathbb{R}^4$ are always Willmore surfaces in $\mathbb{R}^3$. However, Pinkall exhibited in [22] an infinite series of compact embedded Willmore surfaces that are not stereographic projections of compact embedded minimal surfaces in $\mathbb{S}^3$. Yet Kusner conjectured [17] that stereographic projections of Lawson's g-holed tori in $\mathbb{S}^3$ should be global minimizers of $\mathcal{W}$ among all genus g surfaces. This conjecture is still open, except of course for the case $g = 0$ where the round sphere is known to be the unique global minimizer.

The existence of smooth surfaces that minimize the Willmore energy spanning a given boundary and a conormal field has been proved by Schätzle in [27]. Following the notations in [27], we consider a smooth embedded closed oriented curve $\Gamma \subset \mathbb{R}^N$ together with a smooth unit normal field $n_\Gamma \in N_\Gamma$ and we denote as $\pm\Gamma$ and $\pm n_\Gamma$ their possible orientations. We assume that there exist oriented extensions of $\pm\Gamma$, $\pm n_\Gamma$, that is, there are compact oriented surfaces Σ_-, $\Sigma_+ \subset \mathbb{R}^N$ with boundary $\partial\Sigma_\pm = \pm\Gamma$ and conormal vector field $\mathrm{co}_{\Sigma_\pm} = \pm n_\Gamma$ on $\partial\Sigma_\pm$. We also assume that there exists a bounded open set $B \supset \Gamma$ such that the set

$$\{\Sigma_\pm \text{ oriented extensions of } (\Gamma, n_\Gamma),\ \Sigma_+ \text{ connected },\ \Sigma_+ \cup \Sigma_- \subset B,\ \mathcal{W}(\Sigma_+ \cup \Sigma_-) < 8\pi\}$$

is not empty. The condition on energy ensures that $\Sigma_+ \cup \Sigma_-$ is an embedding.

It follows from [27], Corollary 1.2, that the Willmore boundary problem associated with (Γ, n_Γ) in B has a solution, i.e., there exists a compact, oriented, connected, smooth surface $\Sigma \subset B$ with $\partial\Sigma = \Gamma$, $\mathrm{co}_\Sigma = n_\Gamma$ on $\partial\Sigma$, and

$$W(\Sigma) = \min\{W(\tilde{\Sigma}),\ \tilde{\Sigma} \text{ smooth},\ \tilde{\Sigma} \subset B,\ \partial\tilde{\Sigma} = \Gamma,\ \mathrm{co}_{\tilde{\Sigma}} = n_\Gamma \text{ on } \partial\tilde{\Sigma}\}$$

There have been many contributions to the numerical simulation of Willmore surfaces in space dimension $N = 3$. Among them, Hsu, Kusner and Sullivan have tested experimentally in [16] the validity of Kusner's conjecture: starting from a triangulated polyhedron in $\mathbb{R}^3$ that is close to a Lawson's surface of genus g, they let it evolve by a discrete Willmore flow using Brakke's Surface Evolver [6] and check that the solution obtained after convergence is $\mathcal{W}$-stable. Recent updates that Brakke brought to its program give now the possibility to test the flow with various discrete definitions of the mean curvature. Mayer and Simonett [19] introduce a finite difference scheme to approximate axisymmetric solutions of the Willmore flow. Rusu [26] and Clarenz et al. [8] use a finite elements approximation of the flow to compute the evolution of surfaces with or without boundary. In both works, position and mean curvature vector are taken as independent variables, which is also the case of the contribution by Verdera et al. [33], where a triangulated surface with a hole in it is restored using the following approach: by the coarea formula, the Willmore energy (actually a generalization to other curvature exponents) is replaced with the energy of an implicit and smooth representation of the surface, and the mean curvature term is replaced by the divergence of an unknown field that aims to represent the normal field. Droske and Rumpf [9] propose a finite element approach to the Willmore flow but replace the standard flow equation by its level set formulation. The contribution of Dziuk [10] is twofold: it provides a finite element approximation to the Willmore flow with or without boundary conditions that can handle as well embedded or immersed surfaces (turning the surface problem into a quasi-planar problem), and a consistency result showing the convergence of both the discrete surface and the discrete Willmore energy to the continuous surface and its energy when the approximated surface has enough regularity. Bobenko and Schröder [4] use a difference strategy: they introduce a discrete notion of mean curvature for triangulated surfaces computed from the circles circumscribed to each triangle that shares with the continuous definition a few properties, in particular the invariance with respect to the full Möbius group in $\mathbb{R}^3$. This discrete definition is vertex-based and a discrete flow can be derived. Based also on several axiomatic constraints but using a finite elements framework, Wardetzky et al. [34] introduce an edge-based discrete Willmore energy for triangulated surfaces. Olischläger and Rumpf [21] introduce a two step time discretization of the Willmore flow that extends to the Willmore case, at least formally, the discrete time approximation of the mean curvature motion due to Almgren, Taylor, and Wang [2], and Luckhaus and Sturzenhecker [18]. The strategy consists in using the mean curvature flow to compute an approximation of the mean curvature and plugging it in a time discrete approximation of the Willmore flow. Grzibovskis and Heintz [14], and Esedoglu et al. [11] discuss how 4th order flows can be approximated by iterative convolution with suitable kernels and thresholding.

While all the previous approaches yield approximations of critical points of the Willmore energy, our motivation in this paper is to approximate global minimizers. This is an obviously nontrivial task due to the high nonlinearity and nonconvexity of the energy. Yet, for the simpler area functional, Sullivan [31] has shown with a calibration argument that the task of finding minimal surfaces can be turned into a linear problem. Even more, when a discrete solution is seeked among surfaces that are union of faces in a cubic grid partition of $\mathbb{R}^3$, he proved that the minimization of the linear program is equivalent to solving a minimum-cost circulation network flow problem, for which efficient codes have been developed by Boykov and Kolmogorov [5] after Ford and Fulkerson [12]. Sullivan [31] did not provide experiments in his paper but this was done recently by Grady [13], with applications to the segmentation of medical images.

The linear formulation that we propose here is based on two key ideas: the concept of surface continuation constraints that has been pioneered by Sullivan [31] and Grady [13], and the representation of a triangular surface using pairs of triangles. With this representation and a suitable definition of discrete mean curvature, we are able to turn into a linear formulation the task of minimizing discrete representations of any functional of the form

$$W_\varphi(\Sigma) = \int_\Sigma \varphi(x, n, H)dA$$

among discrete immersed surfaces with boundary constraints:

$$\partial\Sigma = \Gamma, \quad \mathrm{co}_{\tilde{\Sigma}} = n_\Gamma \text{ on } \partial\Sigma.$$

In the expression of $W_\varphi(\Sigma)$, x denotes the space variable, n the normal vector field on Σ and H the mean curvature vector. The linear problem we obtain involves integer-valued unknowns and does not seem to admit any simple graph-based equivalent. We will therefore discuss whether classical strategies for linear optimization can be used.

The paper is organized as follows: in Section 2 we discuss both the chosen representation of surfaces and the definition of discrete mean curvature. In Section 3 we present a first possible approach yielding a quadratic energy. We present in Section 4 our linear formulation and discuss whether it can be tackled by classical linear optimization techniques.

2 Discrete Framework

2.1 Triangular Meshes from a Set of Pre-defined Triangles

The equivalence shown by Sullivan between finding minimal surfaces and solving a flow problem holds true for discrete surfaces defined as a connected set of cell faces in a cellular complex discrete representation of the space. We will consider here polyhedral surfaces defined as union of triangles with vertices in (a finite subset of) the cubic lattice $\epsilon\mathbb{Z}^3$ where $\epsilon = \frac{1}{n}$ is the resolution scale. Not all possible triangles are allowed but only those respecting a specified limit on the maximal edge length. We assume that each triangle, as well as each triangle edge, is represented twice, once for each orientation. We let $\mathcal{I}$ denote the collection of oriented triangles, $N = |\mathcal{I}|$ its cardinality, and M the number of oriented triangle edges. The constrained boundary is given as a contiguous oriented set of triangle edges. The orientation of the boundary constrains the spanning surfaces since we will allow only spanning triangles whose orientation is compatible.

In this framework, one can represent a triangular mesh as a binary indicator vector $x = \{0,1\}^N$ where 1 means that the respective triangle is present in the mesh, 0 that it is not. Obviously, not all binary indicator vectors can be associated with a triangular surface since the corresponding triangles may not be contiguous. However, as discussed by Grady [13] and, in a slightly different setting, by Sullivan [30,31], it is possible to write in a linear form the constraint that only binary vectors that correspond to surfaces spanning the given boundary are considered. We will see that using the same approach here turns the initial boundary value problem into a quadratic program. Another formulation will be necessary to get a linear problem.

2.2 Admissible Indicator Vectors: A First Attempt

To define the set of admissible indicator vectors, we first consider a relationship between oriented triangles and oriented edges which is called *incidence*: a triangle is positive incident to an edge if the edge is one of its borders and the two agree in orientation. It is negative incident if the edge is one of its borders, but in the opposite orientation. Otherwise it is not incident to the edge. For example, the triangle in Figure 1 is positive incident to the edge e_1, negative incident to e_2 and e_3 and not incident to e_4.

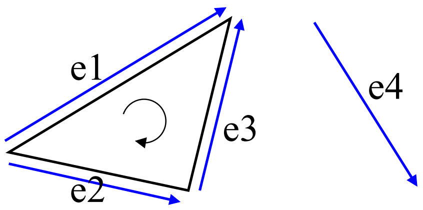


Fig. 1. Incidence of oriented triangles and edges. e_1 is positively incident to the oriented triangle, e_2 and e_3 are negatively incident, and e_4 is not incident to the triangle.

Being defined as above the set of N oriented triangles and their M oriented edges, we introduce the matrix $B = (b_{ij})_{\substack{i\in\{1,\cdots,N\}\\ j\in\{1,\cdots,M\}}}$ whose element b_{ij} gives account of the incidence between triangle i and edge j. More precisely

$$b_{ij} = \begin{cases} 1 & \text{if edge } i \text{ is an edge of triangle } j \text{ with same orientation} \\ -1 & \text{if edge } i \text{ is an edge of triangle } j \text{ with opposite orientation} \\ 0 & \text{otherwise.} \end{cases}$$

The knowledge of which edges are present in the set of prescribed boundary segments is expressed as a vector $r \in \{-1, 0, 1\}^M$ with

$$r_j = \begin{cases} 1 & \text{if the oriented boundary contains the edge } j \\ & \qquad \text{with agreeing orientation} \\ -1 & \text{if the oriented boundary contains the edge } -j \\ & \qquad \text{with opposing orientation} \\ 0 & \text{otherwise.} \end{cases}$$

With these notations, we can now describe the equation system defining that a vector $x \in \{0, 1\}^N$ encodes an oriented triangular mesh with the pre-specified oriented boundary. This system has one equation for each edge. If the edge is not contained in the given boundary, this equation expresses that, among all triangles indicated by x that contain the edge, there are as many triangles with same orientation as the edge as triangles with opposite orientation. If the edge is contained in the boundary with coherent orientation, there must be one more positive incident triangle than negative incident. If it is contained with opposite orientation, there is one less positive than negative incident. Altogether the constraint for edge j can be expressed as the linear equation

$$\sum_i b_{ij}\, x_i = r_j$$

and the entire system as

$$B\, x = r. \tag{1}$$

So far, we did not incorporate the conormal constraint. Actually not all conormal constraints are possible, exactly like not all discrete curves can be spanned in our framework but only union of edges of dictionary triangles, i.e. the collection of triangles defined in the previous section that determine the possible surfaces. For the conormal constraint, only the conormal vectors that are tangent to dictionary triangles sharing an edge with the boundary curve are allowed. Then the conormal constraint can be easily plugged into our formulation by simply imposing the corresponding triangles to be part of the surface, see Figure 2, and by defining accordingly a new boundary indicator vector $\tilde{r}$.

Denoting as $\mathcal{J}$ the collection of those additional triangles, the complete constraint reads

$$\begin{cases} B\, x = \tilde{r} \\ x_j = 1, \; j \in \mathcal{J} \end{cases} \tag{2}$$

We discuss in the next section how discrete mean curvature can be evaluated in this framework.

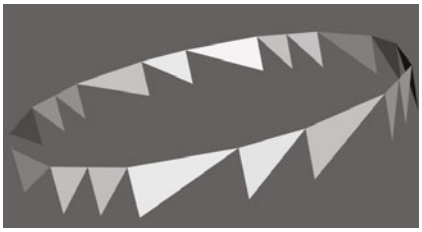


Fig. 2. The boundary and conormal constraints can be imposed by pre-specifying suitable triangles to be part of the surface

2.3 Discrete Mean Curvature on Triangular Meshes

The various definitions of discrete mean curvature that have been proposed in the literature obviously depend on the chosen discrete representations of surfaces. Presenting and discussing all possible definitions is out of the scope of the present paper. The important thing to know is that there is no fully consistent definition: the pointwise convergence of mean curvature cannot be guaranteed in general but only in specific situations [15,20]. Among the many possible definitions, we will use the edge-based one proposed by Polthier [24] for it suits with our framework. Recalling that, in the smooth case but also for generalized surfaces like varifolds [29], the first variation of the area can be written in terms of the mean curvature, the definition due to Polthier of the mean curvature vector at an interior edge e of a simplicial surface reads

$$H(e) = |e| \cos \frac{\theta_e}{2} N_e \tag{3}$$

where $|e|$ is the edge-length, θ_e is the dihedral angle between the two triangles adjacent to e, and N_e is the angle bisecting unit normal vector, i.e., the unit vector collinear to the half sum of the two unit vectors normal to the adjacent triangles (see Figure 3). We remark that this formula is a discrete counterpart of the continuous $H = \kappa_1 + \kappa_2$ depending on the principal curvatures, which is used in many papers for simplicity as definition of mean curvature. When the correct continuous definition $H = \frac{1}{2}(\kappa_1 + \kappa_2)$ is used, the formulas above and hereafter should be adapted. The justification of formula (3) by Polthier [24,25] is as follows: it is exactly the gradient at any point $m \in e$ of the area of the two triangles T_1 and T_2 adjacent to e, and this gradient does not depend on the exact position of m. Indeed, one can subdivide T_1, T_2 in four triangles T_i', $i \in \{1, \cdots, 4\}$ having $m \in e$ as a vertex and such that $T_1 = T_1' \cup T_2'$ and $T_2 = T_3' \cup T_4'$. The area of each triangle is half the product of the opposite edge's length and the height. Therefore, if e_i is the positively oriented edge opposite to m in the triangle T_i' and J_1, J_2 the rotations in the planes of T_1, T_2 by $\frac{\pi}{2}$, the area gradients of T_i', $i \in \{1, \cdots, 4\}$ at m are $\frac{1}{2}J_1e_1$, $\frac{1}{2}J_1e_2$, $\frac{1}{2}J_2e_3$, $\frac{1}{2}J_2e_4$. The sum is the total area gradient of $T_1 \cup T_2$ at m and equals $\frac{1}{2}(J_1e + J_2e)$, which coincides with the formula above.

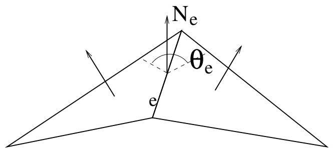


Fig. 3. The edge-based definition of a discrete mean curvature vector due to Polthier [24] depends on the dihedral angle θ_e and the angle bisecting unit normal vector N_e

As discussed by Wardetsky et al. using the Galerkin theory of approximation, this discrete mean curvature is an integrated quantity: it scales as λ when each space dimension is rescaled by λ. A pointwise discrete mean curvature rescaling as $\frac{1}{\lambda}$ is given by (see [34])

$$H^{\text{pw}}(e) = \frac{3|e|}{A_e} \cos\frac{\theta_e}{2} N_e,$$

where A_e denotes the total area of the two triangles adjacent to e. The factor 3 comes from the fact that, when the mean curvatures are summed up over all edges, the area of each triangle is counted three times, once for each edge. Then a discrete counterpart of the energy $\int_{\Sigma} \varphi(H)\, dA$ is given by

$$\sum_{\text{edges } e} \frac{A_e}{3} \varphi(\frac{3|e|}{A_e} \cos\frac{\theta_e}{2} N_e). \tag{4}$$

In particular, the edge-based total squared mean curvature is

$$\sum_{\text{edges } e} \frac{3|e|^2}{A_e} (\cos\frac{\theta_e}{2})^2. \tag{5}$$

3 A Quadratic Program for the Minimization of the Discrete willmore Energy

Ultimately we are aiming at casting the optimization problem in a form that can be handled by standard linear optimization software. Having in mind the framework described above where a discrete surface spanning the prescribed discrete boundary is given as a collection of oriented triangles satisfying equation (2) and chosen among a pre-specified collection of triangles, a somewhat natural direction at first glance seems to be solving a *quadratic program*. Like in Section 2.1, let us indeed denote as (x_i) the collection of binary variables associated to the "dictionary" of triangles (T_i) and define

- e_{ij} the common edge to two adjacent triangles T_i and T_j;
- θ_{ij} the corresponding dihedral angle;

– N_{ij} the angle bisecting unit normal;
– A_{ij} the total area of both triangles.

Then a continuous energy of the form $\int_\Sigma \varphi(x,n,H)dA$ can be discretized as

$$\sum_{i,j} q_{ij}\, x_i\, x_j \tag{6}$$

with $$q_{ij} = \begin{cases} \frac{1}{2}\frac{A_{ij}}{3}\varphi(e_{ij}, N_{ij}, \frac{3|e_{ij}|}{A_{ij}}\cos\frac{\theta_{ij}}{2}N_{ij}) & \text{if } i \neq j \text{ are adjacent} \\ \tilde{\varphi}(T_i, N_i) & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases}$$

where $\tilde{\varphi}$ allows to incorporate dependences on each triangle T_i's position and unit normal N_i. In particular, the discrete Willmore energy is

$$\sum_{i,j} q^w_{ij} x_i\, x_j \tag{7}$$

with

$$q^w_{ij} = \begin{cases} \frac{3|e_{ij}|^2}{2A_{ij}}(\cos\frac{\theta_{ij}}{2})^2 & \text{if } i \neq j \text{ are adjacent} \\ 0 & \text{otherwise.} \end{cases}$$

Assuming that the maps φ and $\tilde{\varphi}$ are positive-valued, both energy matrices $Q = (q_{ij})$ and $Q^w = (q^w_{ij})$ are symmetric matrices in $\mathbb{R}^{+N\times N}$, and the minimization of either (6) or (7) with boundary constraints turns out to be the following quadratic program with linear and integrality constraints:

$$\begin{aligned} \min_x \quad & \langle Q\, x, x\rangle \\ \text{such that } & B\, x = r \\ & x_i = 1 \;\; \forall i \in \mathcal{J} \\ & x \in \{0,1\}^N \qquad . \end{aligned}$$

We know of no solution to solve this problem efficiently due to the integrality constraint. What is worse, even the relaxed problem where one optimizes over $x \in [0,1]^N$ is very hard to solve: terms of the form $x_i x_j$ with $i \neq j$ are indefinite, so (unless Q has a dominant diagonal) the objective function is a non-convex one.

Moreover, a solution to the relaxed problem would not be of practical use: already for the 2D-problem of optimizing curvature energies over curves in the plane, the respective quadratic program favors fractional solutions. The relaxation would therefore not be useful for solving the integer program. However, in this case Amini et al. [3] showed that one can solve a linear program instead. This inspired us for the major contribution of this work: to cast the problem as an integer linear program.

4 An Integer Linear Programming Approach

4.1 Augmented Indicator Vectors

The key idea of the proposed integer linear program is to consider additional indicator vectors. Aside from the indicator variables x_i for basic triangles, one now also considers entries x_{ij} corresponding to *pairs* of adjacent triangles[1]. Such a pair is called *quadrangle* in the following. We will denote $\hat{x}$ the augmented vector $(x_1, \cdots, x_N, \cdots, x_{ij}, \cdots)$ where $i \neq j$ run over all indices of adjacent triangles. The cost function can be easily written in a linear form with this augmented vector, i.e. it reads

$$\sum w_k \hat{x}_k$$

with (see the notations of the previous section)

$$w_k = \begin{cases} q_{ii} \text{ if } \hat{x}_k = x_i \\ q_{ij} \text{ if } \hat{x}_k = x_{ij}. \end{cases}$$

The major problem to overcome is how to set up a system of constraints that guarantees consistency of the augmented vector: the indicator variable x_{ij} for the pair of triangles i and j should be 1 if and only if both the variables x_i and x_j are 1. Otherwise it should be 0. In addition, one again wants to optimize only over indicator vectors that correspond to a triangular mesh.

To encode this in a linear constraint system, a couple of changes are necessary. First of all, we will now have a constraint for each pair of triangle and adjacent edge. Secondly, edges are no longer oriented. Still, the set of pre-specified indices $\mathcal{J}$ implies that the orientation of the border is fixed - we still require that for each edge of the boundary an adjacent (oriented) triangle is fixed to constrain the conormal information.

To encode the constraint system we introduce a modified notion of incidence. We are no longer interested in incidence of triangles and edges. Instead we now consider the incidence of both triangles and quadrangles to pairs of triangles and (adjacent) edges.

For convenience, we define that triangles are positive incident to a pair of edge and triangle, whereas all quadrangles are negative incident.

We propose an incidence matrix where lines correspond to pairs (triangle, edge) and columns to either triangles or quadrangles. The entries of this incidence matrix are either the incidence of a pair (triangle, edge) with a triangle, defined as

$$d((\text{triangle } k, \text{edge } e), \text{triangle } i) = \begin{cases} 1 & \text{if } i = k,\ e \text{ is an edge of triangle } i \\ 0 & \text{otherwise,} \end{cases}$$

[1] This strategy of doubling the variables shares some similarity with techniques in semi-definite programming.

or the incidence of a pair (triangle, edge) with a quadrangle, defined as

$$d((\text{triangle } k, \text{edge } e), \text{quadrangle } ij) = \begin{cases} -1 & \text{if } i{=}k \text{ or } j{=}k \text{ and } i, j \text{ share } e \\ 0 & \text{otherwise.} \end{cases}$$

The columns of this incidence matrix are of two types: either with only 0's and exactly three 1 (a column corresponding to a triangle T, whose three edges are found at lines (T, e_1), (T, e_2), (T, e_3)), or with only 0's and exactly two (-1)'s (a column corresponding to a quadrangle (T_1, T_2) that matches with lines (T_1, e_{12}) and (T_2, e_{12})).

Again, both the conormal constraints and the boundary edges can be imposed by imposing additional triangles indexed by a collection $\mathcal{J}$ of indices. The general constraint has the form

$$\sum_i d((x_k, e), x_i) + \sum_{i,j} d((x_k, e), x_{ij}) = r'_{(k,e)},$$

where the right-hand side depends whether the edge e is shared by two triangles of the surface (and even several quadrangles in case of self-intersection), or belongs to the new boundary indicated by the additional triangles. If e is an inner edge, then the sum must be zero due to our definition of d, otherwise there is an adjacent triangle, but no adjacent quadrangle, so the right-hand side should be 1:

$$r'_{(k,e)} = \begin{cases} 1 & \text{if } k \in \mathcal{J}, e \text{ is part of the modified boundary} \\ 0 & \text{otherwise.} \end{cases}$$

To sum up, we get the following integer linear program:

$$\begin{aligned} \min_{\hat{x}} \quad & \langle w, \hat{x} \rangle \\ \text{such that } & D\,\hat{x} = r' \\ & \hat{x}_j = 1 \quad \forall j \in \mathcal{J} \\ & \hat{x}_i \in \{0, 1\} \quad \forall i \in \{1, \dots, \hat{N}\} \end{aligned} \tag{8}$$

where $\hat{N}$ is the total number of entries in $\hat{x}$, namely all triangles plus all pairs of adjacent triangles. It is worth noticing that such formulation allows triangle surfaces with self-intersection.

4.2 On the Linear Programming Relaxation

Solving integer linear programs is an NP-complete problem, see e.g. [28, Chapter 18.1]. This implies that, to the noticeable exception of a few particular problems [28], no efficient solutions are known. As a consequence one often resorts to solving the corresponding linear programming (LP) relaxation, i.e. one drops the integrality constraints. In our case this means to solve the problem:

$$\begin{aligned} \min_{\hat{x}} \quad & \langle w, \hat{x} \rangle \\ \text{such that } & D\,\hat{x} = r' \\ & \hat{x}_j = 1 \quad \forall j \in \mathcal{J} \\ & 0 \leq \hat{x}_i \leq 1 \quad \forall i \in \{1, \dots, \hat{N}\} \end{aligned} \tag{9}$$

or, equivalently, by suitably augmenting D and r' in order to incorporate the second constraint $\hat{x}_j = 1$, $\forall j \in \mathcal{J}$:

$$\min_{\hat{x}} \langle w, \hat{x} \rangle \quad \text{such that} \quad \begin{array}{l} \hat{D}\hat{x} = \hat{r} \\ 0 \leq \hat{x}_i \leq 1 \quad \forall i \in \{1, \dots, \hat{N}\} \end{array} \tag{10}$$

There are various algorithms for solving this problem, the most classical being the simplex algorithm and several interior point algorithms. Let us now discuss the conditions under which these relaxed solutions are also solutions of the original integer linear program. Recalling the basics of LP-relaxation [28], the set of admissible solutions

$$P = \{\hat{x} \in \mathbb{R}^{\hat{N}},\ \hat{D}\hat{x} = \hat{r},\ 0 \leq x \leq 1\}$$

is a polyhedron, i.e. a finite intersection of half-spaces in $\mathbb{R}^{\hat{N}}$. A classical result states that minimizing solutions for the linear objective functions can be sought among the extremal points of P only, i.e. its vertices. Denoting P_e the integral envelope of P, that is the convex envelope of $P \cap \mathbb{Z}^{\hat{N}}$, another classical result states that P has integral vertices only (i.e. vertices with integral coordinates) if and only if $P = P_e$

Since $P = \{\hat{x} \in \mathbb{R}^{\hat{N}},\ \hat{D}\hat{x} = \hat{r}, 0 \leq \hat{x} \leq 1\}$, according to Theorem 19.3 in [28], a *sufficient* condition for having $P = P_e$ is the property of B being totally unimodular, i.e. any square submatrix has determinant either 0, -1 or 1. Under this condition, any extremal point of P that is a solution of

$$\min_{\hat{D}\hat{x} = \hat{r},\, \hat{x}_i \in [0,1]} \langle w, \hat{x} \rangle$$

has integral coordinates therefore is a solution of the original integer linear program

$$\min_{\hat{D}\hat{x} = \hat{r},\, \hat{x}_i \in \{0,1\}} \langle w, \hat{x} \rangle.$$

Theorem 19.3 in [28] mentions an interesting characterization of total unimodularity due to Paul Camion [7]: a matrix is totally unimodular if, and only if, the sum of the entries of every Eulerian square submatrix (i.e. with even rows and columns) is divisible by four.

Unfortunately, we can prove that, as soon as the triangle space is rich enough, the incidence matrix $\hat{D}$ does not satisfy Camion's criterion, therefore is not totally unimodular, and neither are the matrices for richer triangles spaces. As a consequence, there are choices of the triangle space for which the polyhedron

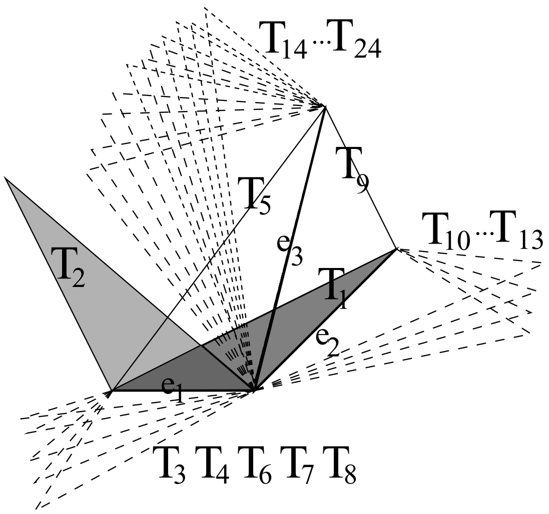


Fig. 4. A configuration in a triangle space with sufficient resolution. The associated incidence matrix is Eulerian (see text) but does not satisfy Camion's criterion, thus is not totally unimodular.

$P = \{\hat{x} \in \mathbb{R}^{\hat{N}},\ \hat{D}\hat{x} = \hat{r}, 0 \leq \hat{x} \leq 1\}$ may have not only integral vertices, or more precisely one cannot guarantee this property thanks to total unimodularity. This is summarized in the following theorem.

Theorem 1. *The incidence matrix associated with any triangle space where each triangle has a large enough number of adjacent neighbors is not totally unimodular.*

Proof. We show in Figure 4 a configuration and, in Table 1, an associated square submatrix of the incidence matrix. The sum of entries over each line and the sum over each column are even, though the total sum of the matrix entries is not divisible by four. By a result of Camion [7], the incidence matrix is not totally unimodular which yields the conclusion according to Theorem 19.3 in [28]. Clearly, any triangle space for which this configuration can occur is also associated to an incidence matrix that is not totally unimodular. □

It is worth noticing that the previous theorem does not imply that the extremal points of the polyhedron P are necessarily not all integral. It only states that this cannot be guaranteed as usual by the criterion of total unimodularity.

For the sake of completeness, let us mention that there actually exist necessary and sufficient conditions of integrality due to Truemper [32], or sufficient conditions different from above due to Grady [13], but we have not been able to exploit them so far.

We will discuss in the next section what additional informations about integrality can be obtained from a few experiments that we have done using classical solvers for addressing the relaxed linear problem.

Table 1. A square incidence matrix associated with the configuration in Figure 4. It is Eulerian, i.e. the sum along each line and the sum along each column are even, but the total sum is not divisible by four. According to Camion [7], the matrix is not totally unimodular.

	T_1	T_5	T_9	$T_{1,2}$	$T_{2,3}$	$T_{3,4}$	$T_{4,5}$	$T_{2,5}$	$T_{5,6}$	$T_{1,6}$	$T_{1,7}$	$T_{7,8}$	$T_{2,8}$	$T_{1,9}$	$T_{9,10}$	$T_{10,11}$	$T_{1,11}$	$T_{1,12}$	$T_{12,13}$	$T_{9,13}$	$T_{9,5}$	$T_{5,14}$	$T_{14,15}$	$T_{9,15}$	$T_{9,16}$	$T_{16,17}$	$T_{5,17}$	$\sum$
(T_1, e_1)	1			-1						-1	-1																	-2
(T_2, e_1)				-1	-1			-1					-1															-4
(T_3, e_1)					-1	-1																						-2
(T_4, e_1)						-1	-1																					-2
(T_5, e_1)		1					-1	-1	-1																			-2
(T_6, e_1)									-1	-1																		-2
(T_7, e_1)											-1	-1																-2
(T_8, e_1)												-1	-1															-2
(T_1, e_2)	1													-1			-1	-1										-2
(T_9, e_2)			1											-1	-1					-1								-2
(T_{10}, e_2)															-1	-1												-2
(T_{11}, e_2)																-1	-1											-2
(T_{12}, e_2)																		-1	-1									-2
(T_{13}, e_2)																			-1	-1								-2
(T_9, e_3)			1																		-1			-1	-1			-2
(T_5, e_3)		1																			-1	-1					-1	-2
(T_{14}, e_3)																						-1	-1					-2
(T_{15}, e_3)																							-1	-1				-2
(T_{16}, e_3)																									-1	-1		-2
(T_{17}, e_3)																										-1	-1	-2
	plus 7 lines $(T_{18}, e_3), \cdots, (T_{24}, e_3)$ with only 0 entries to have a square matrix																											
$\sum$	2	2	2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-42

4.3 Testing the Relaxed Linear Problem

We have tested the relaxed formulation on a few examples at low-resolution using the dual simplex method implemented in the CLP solver. The main reason for using low-resolution is that the number of triangles becomes significantly important as the resolution increases, and both the computational cost and the memory requirements tend to become large. Another reason for working at low-resolution is that there is no need to go high before finding a case of non-integrality. Indeed, consider the examples in Figure 5: integral solutions are obtained when the resolution is very low (i.e. when there is no risk to have configurations like in Figure 4). In the last configuration, however, the optimal solution of the relaxed problem has fractional entries. This confirms that our initial problem cannot be addressed though the classical techniques of relaxation, and with usual LP solvers.

4.4 On Integer Linear Programming

Our results above indicate that, necessarily, integer linear solvers [28,1] should be used. These commonly start with solving the linear programming relaxations, then derive further valid inequalities (called *cuts*) and/or apply a branch-and-bound scheme. Due to the small number of fractional values that we have observed in our experiments, it is quite likely that the derivation of a few cuts only would give integral solutions. However, we did not test this so far because of the running times of this approach: in cases where we get fractional solutions the dual simplex method often needs as long as two weeks and up to 12 GB memory! From experience with other linear programming problems we consider it likely that the interior point methods implemented in commercial solvers will be much faster here (we expect less than a day). At the same time, we expect the memory consumption to be considerably higher, so the method would most probably be unusable in practice.

We strongly believe that a specific integer linear solver should be developed rather than using general implementations. It is well known that, for a few problems like the knapsack problem, see Chapter 24.6 of [28], their specific structure gives rise to ad-hoc efficient approaches. Recalling that our incidence matrix is very sparse and well structured (the nonzero entries of each column are either exactly two (-1), or exactly three 1) we strongly believe that an efficient integer solver can be developed and our approach can be amenable to higher-resolution results in the near future.

5 Conclusion

We have shown that the minimization under boundary constraints of mean curvature based energies over surfaces, and in particular the Willmore energy, can be cast as an integer linear program. Unfortunately, this integer program is not equivalent to its relaxation so the classical LP algorithms offer no warranty

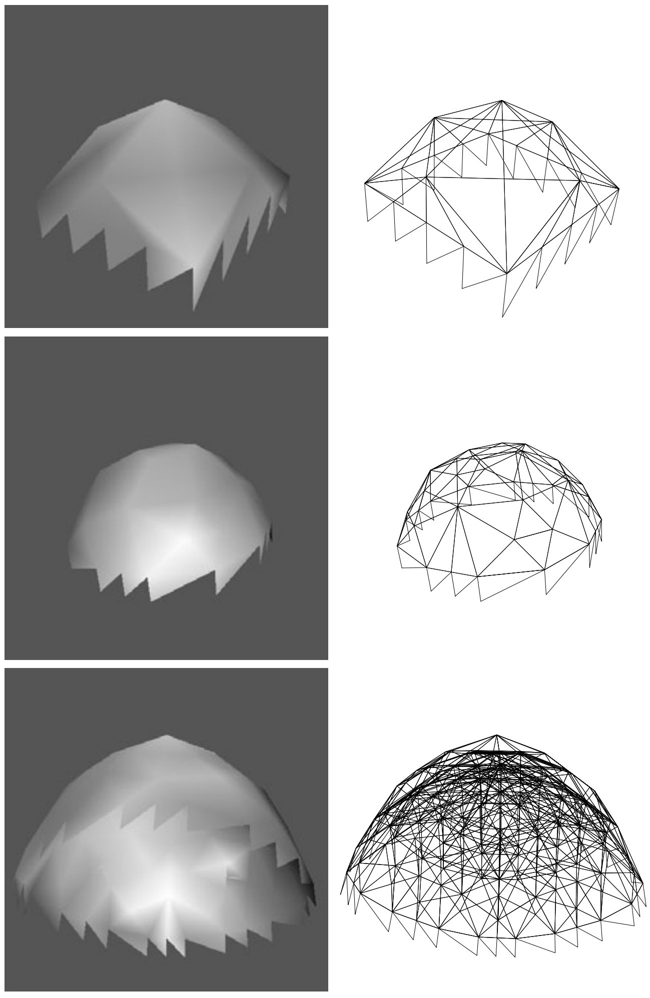


Fig. 5. A series of experiments (the result and the mesh edges) with increasing resolution of the triangle space (and various boundary constraints). An integral solution of the relaxed problem is obtained by a standard LP-solver in both top cases. As for the last case, the triangle space resolution is now large enough for having configurations similar to the counterexample of Figure 4. And indeed, an optimal solution is found for the relaxed problem that is not integral. The mesh on the bottom-right shows actually two nested semi-spheres whose triangles have, at least for a few of them, non binary labels.

that the integer optimal solution will be found. This implies that pure integer linear algorithms must be used, which are in general much more involved. We believe however that the particular structure of the problem paves the way to a dedicated algorithm that would provide high-resolution *global* minimizers of the Willmore boundary problem and generalizations. This is the purpose of future research.

References

1. Achterberg, T.: Constraint Integer Programming. PhD thesis, Technische Universität Berlin (2007)
2. Almgren, F., Taylor, J.E., Wang, L.-H.: Curvature-driven flows: a variational approach. SIAM Journal on Control and Optimization 3, 387–438 (1993)
3. Amini, A.A., Weymouth, T.E., Jain, R.C.: Using dynamic programming for solving variational problems in vision. IEEE Trans. on Patt. Anal. and Mach. Intell. 12(9), 855–867 (1990)
4. Bobenko, A.I., Schröder, P.: Discrete Willmore Flow. In: Eurographics Symposium on Geometry Processing (2005)
5. Boykov, Y., Kolmogorov, V.: An Experimental Comparison of Min-cut/Max-flow Algorithms for Energy Minimization in Vision. In: Figueiredo, M., Zerubia, J., Jain, A.K. (eds.) EMMCVPR 2001. LNCS, vol. 2134, pp. 359–374. Springer, Heidelberg (2001)
6. Brakke, K.A.: The surface evolver. Experimental Mathematics 1(2), 141–165 (1992)
7. Camion, P.: Characterization of totally unimodular matrices. Proc. Am. Math. Soc. 16(5), 1068–1073 (1965)
8. Clarenz, U., Diewald, U., Dziuk, G., Rumpf, M., Rusu, R.: A finite element method for surface restoration with smooth boundary conditions. Computer Aided Geometric Design 21(5), 427–445 (2004)
9. Droske, M., Rumpf, M.: A level set formulation for Willmore flow. Interfaces and Free Boundaries 6(3) (2004)
10. Dziuk, G.: Computational parametric Willmore flow. Numer. Math. 111, 55–80 (2008)
11. Esedoglu, S., Ruuth, S., Tsai, R.Y.: Threshold dynamics for high order geometric motions. Technical report, UCLA CAM report (2006)
12. Ford, L.R., Fulkerson, D.: Flows in Networks. Princeton University Press, Princeton (1962)
13. Grady, L.: Minimal surfaces extend shortest path segmentation methods to 3D. IEEE Trans. on Patt. Anal. and Mach. Intell. 32(2), 321–334 (2010)
14. Grzibovskis, R., Heintz, A.: A convolution-thresholding scheme for the Willmore flow. Technical Report Preprint 34, Chalmers University of Technology, Göteborg, Sweden (2003)
15. Hildebrandt, K., Polthier, K., Wardetzky, M.: On the convergence of metric and geometric properties of polyhedral surfaces. Geometriae Dedicata 123, 89–112 (2005)
16. Hsu, L., Kusner, R., Sullivan, J.: Minimizing the squared mean curvature integral for surfaces in space forms. Experimental Mathematics 1(3), 191–207 (1992)
17. Kusner, R.: Comparison surfaces for the Willmore problem. Pacific J. Math. 138(2) (1989)
18. Luckhaus, S., Sturzenhecker, T.: Implicit time discretization for the mean curvature flow equation. Calculus of Variations and Partial Differential Equations 3(2), 253–271 (1995)

19. Mayer, U.F., Simonett, G.: A numerical scheme for axisymmetric solutions of curvature-driven free boundary problems, with applications to the willmore flow. Interfaces and Free Boundaries 4, 89–109 (2002)
20. Morvan, J.-M.: Generalized Curvatures, 1st edn. Springer Publishing Company, Incorporated (2008)
21. Olischläger, N., Rumpf, M.: Two step time discretization of Willmore flow. In: Proc. 13th IMA Int. Conf. on Math. of Surfaces, pp. 278–292. Springer, Heidelberg (2009)
22. Pinkall, U.: Hopf tori in S^3. Invent. Math. 81, 379–386 (1985)
23. Pinkall, U., Sterling, I.: Willmore surfaces. The Mathematical Intelligencer 9(2) (1987)
24. Polthier, K.: Polyhedral surfaces of constant mean curvature. Habilitation thesis, TU Berlin (2002)
25. Polthier, K.: Computational aspects of discrete minimal surfaces. In: Global Theory of Minimal Surfaces, Proc. of the Clay Mathematics Institute Summer School (2005)
26. Rusu, R.: An algorithm for the elastic flow of surfaces. Interfaces and Free Boundaries 7, 229–239 (2005)
27. Schätzle, R.: The Willmore boundary problem. Calc. Var. Part. Diff. Equ. 37, 275–302 (2010)
28. Schrijver, A.: Theory of linear and integer programming. Wiley-Interscience series in discrete mathematics. John Wiley and Sons, Chichester (1994)
29. Simon, L.: Lectures on Geometric Measure Theory. In: Proc. of the Center for Mathematical Analysis, vol. 3, Australian National University (1983)
30. Sullivan, J.M.: A Crystalline Approximation Theorem for Hypersurfaces. PhD thesis, Princeton University, Princeton (1992)
31. Sullivan, J.M.: Computing hypersurfaces which minimize surface energy plus bulk energy. In: Motion by Mean Curvature and Related Topics, pp. 186–197 (1994)
32. Truemper, K.: Algebraic Characterizations of Unimodular Matrices. SIAM J. Applied Math. 35(2), 328–332 (1978)
33. Verdera, J., Caselles, V., Bertalmio, M., Sapiro, G.: Inpainting surface holes. In: Int. Conference on Image Processing, pp. 903–906 (2003)
34. Wardetzky, M., Bergou, M., Harmon, D., Zorin, D., Grinspun, E.: Discrete quadratic curvature energies. Computer Aided Geometric Design 24(8-9), 499–518 (2007)

Interpolation Function of Generalized $q-$Bernstein-Type Basis Polynomials and Applications

Yilmaz Simsek

Department of Mathematics,
Faculty of Science University of Akdeniz TR-07058 Antalya-Turkey
ysimsek@akdeniz.edu.tr

Abstract. The main aim of this paper is to construct a new generating function for the generalized q-Bernstein-type basis polynomials and to derive fundamental properties of these polynomials. We establish relations between the generalized q-Bernstein-type basis polynomials, the Bernoulli polynomials of higher-order and the generalized Stirling numbers of the second kind. By applying Mellin transform to this generating function, we also construct an interpolating function, which interpolates the generalized q-Bernstein-type basis polynomials at negative integers. Furthermore, we give applications on the generalized q-Bernstein-type basis polynomials and the Bézier curves.

Keywords: Bernstein basis polynomials, Generating function, q- Bernstein basis polynomials, Bernoulli polynomials of higher-order, Stirling numbers of the second kind, interpolating function, Mellin transform, Gamma function, beta function, Bézier curves.

1 Introduction

The Bernstein basis polynomials have many applications in computer-aided geometric design, in approximations of functions, in statistics, in numerical analysis, in q-analysis and in the solution of differential equations and in the other fields. The q-Bernstein basis polynomials were introduced by Phillips [21] and also by Lewanowicz and Woźny [15,16]. Recently, many papers have been published on the (q-) Bernstein basis polynomials and other related subjects, cf. [1]-[34], and see also the references cited in each of these earlier works.

It is also very well known that many kinds of the Bernstein-type basis polynomials have been defined in various different ways. Many of the known identities for the Bernstein basis polynomials are currently derived in the usual way, using either the binomial theorem, the binomial distribution and tricky algebraic manipulations. In this paper, our main motivations are to introduce a generating function for the q-Bernstein basis polynomials and to give main properties of these polynomials. We find *functional equations* and *differential equations* for the generating functions of the Stirling numbers of the second kind and the q-Bernstein basis polynomials. Using these equations, we give some properties for these numbers and polynomials.

We summarize our paper results as follows. In Section 2, we give basic properties for the Bernstein basis polynomials and Bernoulli polynomials. In Section 3, we give some relations related to the Stirling numbers of the second kind and their generating function. In Section 4, we define generating function for the q-Bernstein-type basis polynomials. We give fundamental properties of these polynomials. In Section 5, by using the partial derivative of the generating function, we give a recursive relation and the derivative of the q-Bernstein-type basis polynomials. In Section 6, we give relations between the generalized Bernoulli polynomials, the generalized Stirling numbers of the second kind and the q-Bernstein-type basis polynomials. In Section 7, we integral representation of the q-Bernstein-type basis polynomials. In Section 8, by applying Mellin transform to the generating function of the q-Bernstein-type basis polynomials, we construct a complex rational function, related to q-calculus, beta function, and gamma function, which interpolates the q-Bernstein-type polynomials at negative integers. These values are given explicitly in Theorem 12. In Section 9, we propose a modified version of the Bézier curves based on the q-Bernstein-type basis polynomials.

2 Basic Properties of the Classical Bernstein Basis Polynomials, Bernoulli Polynomials and Stirling Numbers of the Second Kind

Here, we collect some well-known definitions and formulas for the Bernstein basis polynomials and the Bernoulli polynomials. It is well known that the Bernstein polynomials play very important role in theory of the Bézier curves, which are of fundamental importance for computer aided geometric design.

Let n be a natural number. Let f be a function on $[0,1]$. The Bernstein operator $\mathbf{T}_n f$ is defined by

$$\mathbf{T}_n f(x) = \sum_{j=0}^{n} f\left(\frac{j}{n}\right) B_j^n(x),$$

where $0 \leq x \leq 1$ and $B_j^n(x)$ denotes the Bernstein basis polynomials of degree n, which are defined by

$$B_j^n(x) = \binom{n}{j} x^j (1-x)^{n-j}, \tag{1}$$

where

$$\binom{n}{j} = \frac{n!}{j!\,(n-j)!}.$$

If $f : [0,1] \to \mathbb{C}$ is a continuous function, the sequence of the Bernstein basis polynomials $\mathbf{T}_n f(x)$ converges uniformly to f on $[0,1]$, cf. ([1]-[31]), and see also the references cited in each of these earlier works.

The Bernstein polynomials are symmetric in the sense that

$$B_j^n(x) = B_{n-j}^n(1-x).$$

These basis polynomials also satisfy

$$B_j^n(0) = \delta_{j,0}$$

and

$$B_j^n(1) = \delta_{j,n},$$

where $\delta_{j,n}$ is a Kronecker delta. For $0 \leq x \leq 1$, the Bernstein basis polynomials are bounded on the interval $[0,1]$:

$$0 \leq B_j^n(x) \leq 1.$$

The Bernoulli polynomials of higher-order, $\mathfrak{B}_n^{(k)}(x)$ are defined by means of the following generating function

$$\mathbf{f}^{(k)}(x,t) = e^{tx}\left(\frac{t}{e^t-1}\right)^k = \sum_{n=0}^{\infty} \mathfrak{B}_n^{(k)}(x)\frac{t^n}{n!}. \tag{2}$$

The polynomials $\mathfrak{B}_n^{(k)}(x)$ are used for many branches of Mathematics, for example in finite differences, in number theory, and as the coefficients in all the usual central difference formulas for interpolation.

Note that taking $k = 1$ in (2), $\mathfrak{B}_n^{(1)}(x)$ denote the usual Bernoulli polynomials, see [17,25,27].

The Stirling numbers of the second kind, $S(n,k)$ are defined by means of the generating function cf. ([5,19,26])

$$F_S(t,k) = \frac{(-1)^k}{k!}(1-e^t)^k = \sum_{n=0}^{\infty} S(n,k)\frac{t^n}{n!}. \tag{3}$$

These numbers play an important role in many branches of Mathematics, for example, in combinatorics, in number theory, in discrete probability distributions for finding higher order moments. In [12], Joarder *et al.* demonstrated the application of the Stirling numbers of the second kind in calculating moments of some discrete distributions, which are the binomial distribution, the geometric distribution and the negative binomial distribution. These numbers satisfy the following relations:

Let n be a nonnegative integer. Let $A_n = \{1, 2, \cdots, n\}$. $S(n,k)$ denotes the number of partitions of A_n into k blocks. For positive integer n,

$$S(n,0) = 0.$$

For nonnegative integer n,

$$S(n,1) = S(n,n) = 1,$$

and

$$S(n,n-1) = \binom{n}{2}.$$

3 Some Properties of the Generalized Stirling Numbers of the Second Kind

Because of many applications of the Stirling numbers, for example in combinatorics, statistics and many branches of Applied Mathematics and Mathematical Physics, there are many generalizations and extensions of these numbers.

Toscano [33], and also Cakic *at al.* [5] defined the generalized Stirling numbers of the second kind, $\mathcal{S}(n,k;y)$ by means of the generating function

$$F_{\mathcal{S}}(y,t,k)=\frac{(-1)^k}{k!}e^{yt}(1-e^t)^k=\sum_{n=0}^{\infty}\mathcal{S}(n,k;y)\frac{t^n}{n!}. \tag{4}$$

Observe that if $y=0$, then it is easy to see that (4) reduces to (3).

Using equation (4), we obtain

$$\frac{(-1)^k}{k!}\left(\sum_{n=0}^{\infty}y^n\frac{t^n}{n!}\right)\left(\sum_{j=0}^{k}\binom{k}{j}(-1)^{k-j}\sum_{n=0}^{\infty}j^n\frac{t^n}{n!}\right)=\sum_{n=0}^{\infty}\mathcal{S}(n,k;y)\frac{t^n}{n!}.$$

By using the Cauchy product in the above equation, after some elementary calculations, we arrive at the following theorem:

Theorem 1. *The following identity holds:*

$$\mathcal{S}(n,k;y)=\frac{1}{k!}\sum_{m=0}^{n}\binom{n}{m}y^n\sum_{j=0}^{k}\binom{k}{j}(-1)^j\left(\frac{j}{y}\right)^m,$$

where

$$0^m=\begin{cases}0 \ if \ m\neq 0,\\ 1 \ if \ m=0.\end{cases}$$

Remark 1. Substituting $y=0$ into Theorem 1, we have

$$\mathcal{S}(n,k;0)=S\left(n,k\right).$$

By (4) and (3), we get the following functional equation:

$$F_{\mathcal{S}}(y,t,k)=e^{yt}F_S(t,k). \tag{5}$$

By the above functional equation, we obtain

$$\left(\sum_{n=0}^{\infty}S(n,k)\frac{t^n}{n!}\right)\left(\sum_{n=0}^{\infty}y^n\frac{t^n}{n!}\right)=\sum_{n=0}^{\infty}\mathcal{S}(n,k;y)\frac{t^n}{n!}.$$

By using Cauchy product in the above equation, we arrive at the following result.

Theorem 2. *The following identity holds:*

$$\mathcal{S}(n,k;y)=\sum_{j=0}^{n}\binom{n}{j}y^{n-j}S(j,k).$$

Remark 2. $\mathcal{S}(n,k;y)$ is a polynomial of degree n.

Differentiating (4) with respect to x and t, we obtain the following partial differential equations:

$$\frac{\partial}{\partial y}F_{\mathcal{S}}(y,t,k)=tF_{\mathcal{S}}(y,t,k),$$

and

$$\frac{\partial}{\partial t}F_{\mathcal{S}}(y,t,k)=yF_{\mathcal{S}}(y,t,k)+F_{\mathcal{S}}(y+1,t,k-1).$$

By using the above equations, we obtain the following theorem:

Theorem 3. *The following derivative and recurrence relation hold, respectively:*

$$\frac{d}{dy}\mathcal{S}(n,k;y)=n\mathcal{S}(n-1,k;y),$$

and

$$\mathcal{S}(n,k;y)=y\mathcal{S}(n-1,k;y)+\mathcal{S}(n-1,k-1;y+1).$$

The Cauchy-type integral of the $\mathcal{S}(n,k;y)$ is given by

$$\mathcal{S}(n,k;y)=\frac{n!}{k!}\frac{1}{2\pi i}\int_{\mathcal{C}}e^{yt}(e^{t}-1)^{k}\frac{dt}{t^{n+1}}, \tag{6}$$

where $\mathcal{C}$ is a small circle around the origin and the integration is in positive direction.

Remark 3. Substituting $y=0$ into (6), we have integral representation of the Stirling numbers of the second kind. Our method is similar to that of Temme's [32].

4 A Generating Function for the Generalized q-Bernstein-Type Basis Polynomials

The aim of this section is to construct a generating function for the q-Bernstein-type basis polynomials. We use the following notation:

$$[x:q]=\frac{1-q^{x}}{1-q}.$$

Observe that

$$\lim_{q\to 1}[x:q]=x.$$

If $q\in\mathbb{C}$, we assume that $\mid q\mid<1$. If $q\in\mathbb{R}$, we assume that $0<q<1$.

We define a generating function for the generalized q-Bernstein-type basis polynomials, $\mathfrak{b}_k^n(x+y;q)$ as follows:

$$\mathcal{F}_{k,q}(t,x;y)=\frac{1}{k!}t^k[x:q]^k e^{[1-x+y:q]t}=\sum_{n=0}^{\infty}\mathfrak{b}_k^n(x+y;q)\frac{t^n}{n!}, \tag{7}$$

where $x,y\in[0,1]$ and k is a nonnegative integer.

By using the Taylor series for $e^{[1-x+y:q]t}$ in (7), we obtain

$$\frac{1}{k!}\sum_{n=0}^{\infty}[x:q]^k\left[1-x+y:q\right]^n\frac{t^{n+k}}{n!}=\sum_{n=0}^{\infty}\mathfrak{b}_k^n(x+y;q)\frac{t^n}{n!}.$$

By comparing the coefficients of t^n on the both sides of the above equation, we deduce one of the main theorems of this paper as follows:

Theorem 4. *Let $x,y\in[0,1]$. Let k and n be nonnegative integers. If $n\geq k$, then*

$$\mathfrak{b}_k^n(x+y;q)=\binom{n}{k}[x:q]^k\left[1-x+y:q\right]^{n-k}.$$

Remark 4. The polynomials $\mathfrak{b}_k^n(x+y;q)$ are called the q-Bernstein-type basis polynomials.

$$\lim_{q\to 1}\mathcal{F}_{k,q}(t,x;y)=F_k(t,x)e^{yt}=\sum_{n=0}^{\infty}\mathfrak{b}_k^n(x+y;1)\frac{t^n}{n!}, \tag{8}$$

where

$$\lim_{q\to 1}\mathfrak{b}_k^n(x+y;q)=\binom{n}{k}x^k(1-x+y)^{n-k}.$$

If we set $y=0$ in (7), we obtain a result given by Simsek at al. [30, Eq-(3.8)]:

$$F_{k,q}(t,x)=\frac{1}{k!}t^k[x:q]^k e^{[1-x:q]t}=\sum_{n=0}^{\infty}Y_n(k,x;q)\frac{t^n}{n!} \tag{9}$$

so that obviously (cf. [1,30]).

$$\lim_{q\to 1}F_{k,q}(t,x)=\frac{1}{k!}t^k x^k e^{(1-x)t}=\sum_{n=k}^{\infty}B_k^n(x)\frac{t^n}{n!}$$

That is

$$\lim_{\substack{q \to 1 \\ y=0}} \mathfrak{b}_k^n(x+y;q) = B_k^n(x),$$

where $B_k^n(x)$ denotes the Bernstein basis polynomials, cf. [1,3,4], [6]-[16], [13]-[24], [28]-[31], [34].

By using (7) and (9), we obtain the following functional equation:

$$\mathcal{F}_{k,q}(t,x;y) = e^{\left(q^{1-x}[y:q]t\right)} F_{k,q}(t,x). \tag{10}$$

By using the Taylor series for $e^{[1-x+y:q]t}$ in (10), we get

$$\sum_{n=0}^{\infty} \mathfrak{b}_k^n(x+y;q)\frac{t^n}{n!} = \sum_{n=0}^{\infty} Y_n(k,x;q)\frac{t^n}{n!} \sum_{n=0}^{\infty} \frac{\left(q^{1-x}[y:q]\right)^n t^n}{n!}.$$

By using the Cauchy product in the above equation, we deduce in the next theorem a relationship between $\mathfrak{b}_k^n(x+y;q)$ and $Y_n(k,x;q)$.

Theorem 5. *Let $x, y \in [0,1]$. Let k and n be nonnegative integers with $n \geq k$. Then*

$$\mathfrak{b}_k^n(x+y;q) = \sum_{j=0}^{n} \binom{n}{j} \left(q^{1-x}[y:q]\right)^{n-j} Y_j(k,x;q),$$

where

$$0^n = \begin{cases} 0 \ if \ n \neq 0, \\ 1 \ if \ n = 0. \end{cases}$$

Let f be a continuous function on $[0,1]$. Then we define q-Bernstein-type operator, $\mathbf{T}_{n,q}(f(x))$ as follows:

$$\mathbf{T}_{n,q}(f(x)) = \sum_{k=0}^{n} f\left(\frac{k}{n}\right) \mathfrak{b}_k^n(x+y;q), \tag{11}$$

where $x, y \in [0,1]$, k and n are positive integers with $n \geq k$.

Setting $f(x) = x$ in (11), then we have

$$\mathbf{T}_{n,q}(x) = \sum_{k=0}^{n} \frac{k}{n} \binom{n}{k} [x:q]^k [1-x+y:q]^{n-k}.$$

From the above, we get

$$\mathbf{T}_{n,q}(x) = [x:q] \sum_{k=0}^{n} \mathfrak{b}_{k-1}^{n-1}(x+y;q).$$

Setting $f(x) = x^2$ in (11), we get

$$\mathbf{T}_{n,q}\left(x^2\right) = \frac{[x:q]}{n}\sum_{k=0}^{n} k\mathfrak{b}_{k-1}^{n-1}(x+y;q).$$

We define the following functional equation, which is a fundamental in **subdivision property** for the q-Bernstein basis functions:

$$\mathcal{F}_{k,q}(t,zx;y) = \mathcal{F}_{k,q}([z]\,t,x;y)e^{[1-z+y]t},$$

where $x,y,z \in [0,1]$.

By using the above equation and (7), after some calculations, we arrive at the following theorem:

Theorem 6. *The following identity holds:*

$$\mathfrak{b}_j^n(xz+y;q) = \sum_{k=j}^{n}\mathfrak{b}_j^k(x+y;q)\mathfrak{b}_k^n(z+y;q).$$

Remark 5. We note that our method is similar to that of [29]. Substituting $y=0$, and $q \to 1$ into Theorem 6, we have

$$B_j^n(xz) = \sum_{k=j}^{n} B_j^k(x)B_k^n(z). \tag{12}$$

The above identity is fundamental in subdivision property for the Bernstein basis functions, cf. [29,7,8,9].

5 Recurrence Relations

By using the partial derivative of a function $\mathcal{F}_{k,q}(t,x;y)$ in (7) with respect to t, we have the following partial derivative equation:

$$\frac{\partial}{\partial t}\mathcal{F}_{k,q}(t,x;y) = [x:q]\mathcal{F}_{k-1,q}(t,x;y) + [1-x+y:q]\,\mathcal{F}_{k,q}(t,x;y). \tag{13}$$

We are now ready to give a *recurrence relation* on the q-Bernstein-type basis polynomials by (13) and (7).

Theorem 7. *Let $x,y \in [0,1]$. Let k and n be nonnegative integers with $n \geq k$. The following recurrence relation holds:*

$$\mathfrak{b}_k^n(x+y;q) = [x:q]\mathfrak{b}_{k-1}^{n-1}(x+y;q) + [1-x+y:q]\,\mathfrak{b}_k^{n-1}(x+y;q). \tag{14}$$

By using the partial derivative of a function $\mathcal{F}_{k,q}(t,x;y)$ in (7) with respect to x, we have

$$\frac{\partial}{\partial x}\mathcal{F}_{k,q}(t,x;y) = \frac{t\log(q)}{1-q}\left(q^{1-x+y}\mathcal{F}_{k,q}(t,x;y) - q^x\mathcal{F}_{k-1,q}(t,x;y)\right).$$

By using the above differential equation, we arrive at the following theorem.

Theorem 8. *Let $x, y \in [0,1]$. Let k and n be nonnegative integers with $n \geq k$. We have*

$$\frac{\partial}{\partial x}\mathfrak{b}_k^n(x+y;q) = \frac{\log(q^n)}{q-1}\left(q^x \mathfrak{b}_{k-1}^{n-1}(x+y;q) - q^{1-x+y}\mathfrak{b}_k^{n-1}(x+y;q)\right). \quad (15)$$

Remark 6. If $q \to 1$ and $y=0$, then (14) reduces to a recursive relation on the Bernstein basis polynomials

$$B_k^n(x) = (1-x)B_k^{n-1}(x) + xB_{k-1}^{n-1}(x)$$

and (15) reduces to derivative of the Bernstein basis polynomials

$$\frac{d}{dx}B_k^n(x) = n\left(B_{k-1}^{n-1}(x) - B_k^{n-1}(x)\right).$$

By the *umbral calculus* convention in (7), we get

$$\frac{([x:q]\,t)^k}{k!} = e^{(\mathfrak{b}_k(x+y;q)-[1-x+y:q])t},$$

where $(\mathfrak{b}_k(x+y;q))^n$ is replaced by $\mathfrak{b}_k^n(x+y;q)$. Thus we have

$$\frac{([x:q]\,t)^k}{k!} = \sum_{n=0}^{\infty}(\mathfrak{b}_k(x+y;q) - [1-x+y:q])^n \frac{t^n}{n!}.$$

From the above equation, we deduce the following theorem.

Theorem 9. *Let $x, y \in [0,1]$. Let k and n be nonnegative integers. If $n = k$, then*

$$[x:q]^n = \sum_{j=0}^{n}\binom{n}{j}(-1)^{n-j}[1-x+y:q]^{n-j}\,\mathfrak{b}_k^j(x+y;q).$$

If $n > k$, then

$$\sum_{j=k+1}^{n}\binom{n}{j}(-1)^{n-j}[1-x+y:q]^{n-j}\,\mathfrak{b}_k^j(x+y;q) = 0.$$

6 Relations between the Polynomials $\mathfrak{b}_k^n(x+y;q)$, $\mathfrak{B}_n^{(v)}(x)$ and $\mathcal{S}(n,k;y)$

By using equations (2), (4) and (8), we obtain the following functional equations:

$$\mathcal{F}_{k,1}(t,x;y) = x^k \mathbf{f}^{(k)}(1-x,t)F_{\mathcal{S}}(y,t,k).$$

By using the above equation, we have

$$\sum_{n=0}^{\infty}\mathfrak{b}_k^n(x+y;1)\frac{t^n}{n!} = x^k\sum_{n=0}^{\infty}\mathfrak{B}_n^{(k)}(1-x)\frac{t^n}{n!}\sum_{n=0}^{\infty}\mathcal{S}(n,k;y)\frac{t^n}{n!}.$$

By using the Cauchy product in the above equation, after some calculations, we find a relation between $\mathfrak{b}_k^n(x+y;1)$, $\mathfrak{B}_n^{(k)}(x)$, and $\mathcal{S}(n,k;y)$ given in the following theorem:

Theorem 10. *The following identity holds:*

$$\mathfrak{b}_k^n(x+y;1) = x^k \sum_{j=k}^{n} \binom{n}{j} \mathcal{S}(j,k;y)\mathfrak{B}_{n-j}^{(k)}(1-x),$$

where $\mathfrak{B}_j^{(k)}(x)$ and $S(n,k)$ denote the classical higher-order Bernoulli polynomials and the Stirling numbers of the second kind, respectively.

Remark 7. The generating function $\mathcal{F}_{k,q}(t,x;y)$ is related to the generalized Bernoulli polynomials and the generalized Stirling numbers of the second kind.

Theorem 11. *The following identity holds:*

$$\begin{aligned}&\sum_{j=k}^{n} \binom{n}{j} q^{(1-x)j} [y:q]^j \, Y_{n-j}(k,x;q)\\&= [x:q]^k \sum_{j=k}^{n} \binom{n}{j} \mathfrak{B}_{n-j}^{(k)}([1-x:q])\,\mathcal{S}(j,k;y),\end{aligned}$$

where

$$0^j = \begin{cases} 0 \ if \ j \neq 0, \\ 1 \ if \ j = 0. \end{cases}$$

Remark 8. Substituting $y=0$ into Theorem 11, then we obtain

$$Y_n(k,x;q) = [x:q]^k \sum_{j=k}^{n} \binom{n}{j} \mathfrak{B}_{n-j}^{(k)}([1-x:q])\, S(j,k)$$

cf. [30, Theorem 4.1].

Proof (of Theorem 11). By using equations (2), (3), (10) and Theorem 5, we obtain

$$\begin{aligned}&\sum_{n=0}^{\infty} Y_n(k,x;q)\frac{t^n}{n!} \sum_{n=0}^{\infty} \left(q^{1-x}[y:q]\right)^n \frac{t^n}{n!}\\&= [x:q]^k \sum_{n=0}^{\infty} \left(\sum_{j=0}^{n} \binom{n}{j} \mathfrak{B}_{n-j}^{(k)}([1-x:q])\,\mathcal{S}(j,k;y) \right) \frac{t^n}{n!}.\end{aligned}$$

By using the Cauchy product in the above equation, we get

$$\begin{aligned}&\sum_{n=0}^{\infty} \left(\sum_{j=0}^{n} \binom{n}{j} \left(q^{1-x}[y:q]\right)^j Y_{n-j}(k,x;q) \right) \frac{t^n}{n!}\\&= [x:q]^k \sum_{n=0}^{\infty} \left(\sum_{j=0}^{n} \binom{n}{j} \mathfrak{B}_{n-j}^{(k)}([1-x:q])\,\mathcal{S}(j,k;y) \right) \frac{t^n}{n!}.\end{aligned}$$

By comparing coefficients of t^n in the both sides of the above, we arrive at the result of the theorem. □

We note that recently many researcher have studied on the generating functions of the special numbers and polynomials, cf. [1]-[34]. Phillips [21] constructed the q-Bernstein basis polynomials. He gave many applications of these polynomials, cf. [21,22,23]. Lewanowicz and Woźny [15,16] gave explicitly the dual basis functions for the generalized q-Bernstein basis polynomials in terms of big q-Jacobi polynomials. They introduced many identities related to the dual generalized Bernstein basis polynomials and q-Hahn polynomials. Woźny and Lewanowicz [34] proposed a novel approach to the problem of multi-degree reduction of Bézier triangular patches with prescribed boundary control points. Their solution can be given in terms of bivariate dual discrete Bernstein polynomials. Goldman [7,8,9] gave many applications on the Bernstein basis polynomials and their generating functions and also Bézier curves. Busé and Goldman [4] gave division algorithms for Bernstein polynomials. They gave three division algorithms for the univariate Bernstein polynomials. Gould [10] obtained different relation between the Bernstein polynomials, generalized Bernoulli polynomials and the second kind Stirling numbers. Oruc and Tuncer [19] also found the relation between the q-Bernstein polynomials and the q-Stirling numbers of the second kind.

7 Integral Representation of the Polynomials $\mathfrak{b}_k^n(x+y;q)$

By using the same method as Lopez and Temme [17], we give integral representation of $\mathfrak{b}_k^n(x+y;q)$ as follows:

$$\mathfrak{b}_k^n(x+y;q) = \frac{n!}{2\pi i}\int_{\mathcal{C}} \mathcal{F}_{k,q}(t,x;y)\frac{dt}{t^{n+1}}, \tag{16}$$

where $\mathcal{C}$ is a circle around the origin and the integration is in positive direction. We note that using (7), (16) and Cauchy residue theorem, integral representation of the polynomials $\mathfrak{b}_k^n(x+y;q)$ is easily obtained.

8 Interpolation Function of the q-Bernstein-Type Basis Polynomials

In this section, by applying the Mellin transform to (7), we construct a complex rational function whose values at negative integers are the polynomials $\mathfrak{b}_k^n(x+y;q)$. These values are given explicitly in Theorem 12. It is known that the generating function $\mathcal{F}_{k,q}(t,x;y)$, given by (7), depends on integer parameter k, on real variable x and y and on complex variable q and t. Therefore the properties of this function are closely related to these variables and parameter. By using this function, many properties of the q-Bernstein-type basis polynomials have been given in the above sections.

Let $z \in \mathbb{C}$. By applying the Mellin transform to (7), we get

$$g_q(z,k;x,y) = \frac{1}{\Gamma(z)} \int_0^\infty t^{z-1} \mathcal{F}_{k,q}(-t,x;y)dt,$$

where $\Gamma(z)$ is the Euler gamma function. By using the above equation, we define interpolation function of the polynomials, $\mathfrak{b}_k^n(x+y;q)$ as follows:

Definition 1. *Let $z \in \mathbb{C}$ and $y - x \neq 1$. Then we define*

$$g_q(z,k;x,y) = (-1)^k \frac{\Gamma(z+k)}{\Gamma(k+1)\Gamma(z)} \frac{[x:q]^k}{[1-x+y:q]^{z+k}}, \tag{17}$$

or for $k > 0$ and $\Re(z) > 0$, we define

$$g_q(z,k;x,y) = (-1)^k \frac{1}{kB(z,k)} \frac{[x:q]^k}{[1-x+y:q]^{z+k}},$$

where $B(z,k)$ denotes the beta function.

Remark 9. Observe that if $y - x = 1$, then

$$g_q(z,k;y-1,y) = \infty.$$

For $x = y - 1$, the function $g_q(z,k;x,y)$ is not analytic. If $y = 0$, then (17) reduces to

$$\lim_{q\to 1} g_q(z,k;x,0) = (-1)^k \frac{\Gamma(z+k)}{\Gamma(k+1)\Gamma(z)} \frac{x^k}{(1-x)^{z+k}},$$

where $z \in \mathbb{C}$ and $x \neq 1$. This function gives us an interpolating function for the Bernstein basis polynomials at negative integers.

Theorem 12. *Let n and k be positive integers with $k \leq n$ and $x, y \in [0,1]$. The following identity holds:*

$$g_q(-n,k;x,y) = \mathfrak{b}_k^n(x+y;q).$$

Proof. We assume that n and k are positive integers with $k \leq n$. It is well-known that the Gamma function has simple poles at $z = -n = 0, -1, -2, -3, \cdots$. The residue of $\Gamma(z)$ is

$$Res(\Gamma(z), -n) = \frac{(-1)^n}{n!}. \tag{18}$$

By substituting $z = -n$ into (17) and using (18), we obtain the desired result. □

Remark 10. If we replace z by negative integers, the meromorphic function $g_q(z,k;x,y)$ interpolates the polynomials $\mathfrak{b}_k^n(x+y;q)$. Substituting $y = 0$ into Theorem 12, one can obtain the following result:

$$\lim_{q\to 1} g_q(-n,k;x,0) = B_k^n(x).$$

9 Applications

The Bernstein basis polynomials are used for important applications in many branches of Mathematics and the other sciences, for instance, in computer-aided geometric design, in approximation theory, in probability theory, in statistic theory, in number theory, in the solution of the differential equations, in numerical analysis, constructing Bézier curves, in q-calculus, in operator theory and applications in computer graphics, cf. [1,3,4], [6]-[16], [13]-[24], [28]-[31], [34].

The Bernstein basis polynomials are used to construct Bézier curves. We now give some remarks on the Bézier curves. In [14], Laurent and Sablonnière published an interesting paper on Pierre Bézier and also on Bézier curvs. According to Sederberg [24], engineers used the Bézier curves in terms of the center of mass of a set of point masses. For example, take into consideration the masses m_0, m_1, m_2, and m_3, which are located at points P_0, P_1, P_2 and P_3, respectively. The center of mass of these four point masses is given by

$$P=\frac{m_0P_0+m_1P_1+m_2P_2+m_3P_3}{m_0+m_1+m_2+m_3},$$

where each mass varies as a function of x cf. [24]. That is $m_0=(1-x)^3$, $m_1=3x(1-x)^2$, $m_2=3x^2(1-x)$ and $m_3=x^3$. For each value of x, the masses assume different weights and their center of mass changes continuously. As x varies between 0 and 1, a curve is swept out by the center of masses. This curve is a cubic Bézier curve. Observe that, for any value of x, this Bézier curve is given by the following relation:

$$P=m_0P_0+m_1P_1+m_2P_2+m_3P_3,$$

where $m_0+m_1+m_2+m_3\equiv 1$ cf. [24]. For $k\in\{0,1,2,3\}$, the masses m_k and the points P_k are called *blending functions* and *control points (Bézier points)*, respectively. The blending functions, in the case of the Bézier curves, are known as the *Bernstein basis polynomials cf. [24].* The Bézier curves are used in computer graphics and related fields and also in the time domain, particularly in animation and interface design, cf. [4,18,24].

By using control points P_0, P_1, P_2,$\cdots$, P_n, the Bézier curve of degree n is defined by

$$\mathbf{P}(x)=\sum_{k=0}^{n}P_kB_k^n(x), \tag{19}$$

where $x\in[0,1]$ and $B_k^n(x)$ denote the Bernstein basis polynomials, cf. [4,18,24].

In [18], Morin and Goldman studied the Bézier subdivision, generating piecewise linear approximations of the Bézier curves that converge to the original Bézier curve. Discrete derivatives of arbitrary order can be associated with these piecewise linear functions by divided differences. They established the convergence of these discrete derivatives to the corresponding continuous derivatives of the initial Bézier curve. They also showed that the control polygons generated by subdivision and degree elevation provide not only an approximation to a Bézier curve, but also approximations of its derivatives of arbitrary order.

Lewanowicz and Woźny [16] constructed a generalized Bézier representation in terms of the generalized q-Bernstein basis polynomials. They gave many aplications related to these polynomials and the Bézier representation.

We may modify the Bézier curves in (19) by introducing the polynomials $\mathfrak{b}_k^n(x+y;q)$ as follows:

$$\mathbf{P}_q(x,y)=\sum_{k=0}^{n}P_k\mathfrak{b}_k^n(x+y;q),$$

with control points P_k, $k\in\{0,1,\cdots,n\}$.

For example, for $n=1$, then

$$\mathbf{P}_q(x,y)=p_0\mathfrak{b}_0^1(x+y;q)+p_1\mathfrak{b}_1^1(x+y;q). \tag{20}$$

That is,

$$\mathbf{P}_q(x,y)=p_0[x:q]+p_1\,[1-x+y:q]$$

is a modified Bézier curve with control points p_0 and p_1.

Remark 11. If $y=0$ and $q\to 1$, then (20) reduces to (19).

Acknowledgement. The author would like to thank the referees for valuable comments and suggestions which improve this present paper. The author is supported by the research fund of Akdeniz University.

References

1. Acikgoz, M., Araci, S.: On generating function of the Bernstein polynomials. In: Numerical Analysis and Applied Mathematics, Amer. Inst. Phys. Conf. Proc., vol. CP1281, pp. 1141–1143 (2010)
2. Berg, S.: Some properties and applications of a ratio of Stirling numbers of the second kind. Scand. J. Statist. 2, 91–94 (1975)
3. Bernstein, S.N.: Démonstration du théorème de Weierstrass fondée sur la calcul des probabilités. Commun. Kharkov Math. Soc. 13, 1–2 (1912)
4. Busé, L., Goldman, R.: Division algorithms for Bernstein polynomials. Comput. Aided Geom. Design. 25, 850–865 (2008)
5. Cakić, N.P., Milovanović, G.V.: On generalized Stirling numbers and polynomials. Math. Balkanica 18, 241–248 (2004)
6. Garloff, J.: The Bernstein expansion and its applications. J. Am. Romanian Acad. 25-27, 80–85 (2003)
7. Goldman, R.: An Integrated Introduction to Computer Graphics and Geometric Modeling. CRC Press, Taylor and Francis (2009)

8. Goldman, R.: Pyramid Algorithms: A Dynamic Programming Approach to Curves and Surfaces for Geometric Modeling. Morgan Kaufmann Publishers, Academic Press (2002)
9. Goldman, R.: Identities for the Univariate and Bivariate Bernstein Basis Functions. In: Paeth, A. (ed.) Graphics Gems V, pp. 149–162. Academic Press, London (1995)
10. Gould, H.W.: A theorem concerning the Bernstein polynomials. Math. Magazine 31, 259–264 (1958)
11. Guan, Z.: Iterated Bernstein polynomial approximations. arXiv:0909.0684v3
12. Joarder, A.H., Mahmood, M.: An inductive derivation of Stirling numbers of the second kind and their applications in Statistics. J. Appl. Math. Decis. Sci. 1, 151–157 (1997)
13. Kowalski, E.: Bernstein polynomials and Brownian motion. Amer. Math. Mon. 113, 865–886 (2006)
14. Laurent, P.-J., Sablonnière, P.: Pierre Bézier: An engineer and a mathematician. Comput. Aided Geom. Design. 18, 609–617 (2001)
15. Lewanowicz, S., Woźny, P.: Generalized Bernstein polynomials. BIT Numer. Math. 44, 63–78 (2004)
16. Lewanowicz, S., Woźny, P.: Dual generalized Bernstein basis. J. Approx. Theory 138, 129–150 (2006)
17. Lopez, L., Temme, N.M.: Hermite polynomials in asymptotic representations of generalized Bernoulli, Euler, Bessel and Buchholz polynomials. Modelling, Analysis and Simulation (MAS), MAS-R9927 September 30, 1–14 (1999)
18. Morin, G., Goldman, R.: On the smooth convergence of subdivision and degree elevation for Bézier curves. Comput. Aided Geom. Design. 18, 657–666 (2001)
19. Oruc, H., Tuncer, N.: On the convergence and Iterates of q -Bernstein polynomials. J. Approx. Theory 117, 301–313 (2002)
20. Ostrovska, S.: The unicity theorems for the limit q-Bernstein opera. Appl. Anal. 88, 161–167 (2009)
21. Phillips, G.M.: Bernstein polynomials based on the q -integers. The heritage of P. L. Chebyshev: a Festschrift in honor of the 70th birthday of T. J. Rivlin, Ann. Numer. Math. 4, 511–518 (1997)
22. Phillips, G.M.: Interpolation and approximation by polynomials, CMS Books in Mathematics/ Ouvrages de Mathématiques de la SMC, vol. 14. Springer, New York (2003)
23. Phillips, G.M.: A survey of results on the q-Bernstein polynomials. IMA J. Numer. Anal. Advance Access 1–12 (June 23, 2009); published online
24. Sederberg, T.: BYU Bézier curves, http://www.tsplines.com/resources/class_notes/Bezier_curves.pdf
25. Simsek, Y.: Twisted (h, q)-Bernoulli numbers and polynomials related to twisted (h, q)-zeta function and L-function. J. Math. Anal. Appl. 324, 790–804 (2006)
26. Simsek, Y.: On q-deformed Stirling numbers. International Journal of Applied Mathematics & Statistics, arXiv:0711.0481v1 (to appear)
27. Simsek, Y., Kurt, V., Kim, D.: New approach to the complete sum of products of the twisted (h, q)-Bernoulli numbers and polynomials. J. Nonlinear Math. Phys. 14, 44–56 (2007)
28. Simsek, Y.: Construction a new generating function of Bernstein type polynomials. Appl. Math. Comput. 218(3), 1072–1076 (2011), doi:10.1016/j.amc.2011.01.074
29. Simsek, Y.: Generating functions for the Bernstein polynomials: A unified approach to deriving identities for the Bernstein basis functions. arXiv:1012.5538v1

30. Simsek, Y., Acikgoz, M.: A new generating function of (q-) Bernstein-type polynomials and their interpolation function. Abstr. Appl. Anal., Article ID 769095, 1–12 (2010)
31. Stancu, D.D.: Asupra unci generalizano a polynomdorlui Bernstein. Studia Univ. Babés-Bolyai 2, 31–45 (1969)
32. Temme, N.M.: Asymptotic estimates of Stirling numbers. Stud. Appl. Math. 89, 233–243 (1993)
33. Toscano, L.: Generalizzazioni dei polinomi di Laguerre e dei polinomi attuariali. Riv. Mat. Univ. Purma. 2, 191–226 (1970)
34. Woźny, P., Lewanowicz, S.: Constrained multi-degree reduction of triangular Bézier surfaces using dual Bernstein polynomials. J. Comput. Appl. Math. 235, 785–804 (2010)

Differential Behaviour of Iteratively Generated Curves

Dmitry Sokolov[1], Christian Gentil[2], and Hicham Bensoudane[3]

[1] LORIA - Alice, Campus Scientifique, BP 239
54506 Vandoeuvre-les-Nancy Cedex, France
dmitry.sokolov@loria.fr
[2] Laboratoire LE2I — UMR CNRS 5158
Faculté des Sciences Mirande, Aile de l'Ingénieur, 21078 DIJON Cedex
christian.gentil@u-bourgogne.fr
[3] Laboratoire LE2I — UMR CNRS 5158
Faculté des Sciences Mirande, Aile de l'Ingénieur, 21078 DIJON Cedex
Hicham.Bensoudane@u-bourgogne.fr

Abstract. The aim of our work is to specify and develop a geometric modeler, based on the formalism of iterated function systems with the following objectives: access to a new universe of original, various, aesthetic shapes, modeling of conventional shapes (smooth surfaces, solids) and unconventional shapes (rough surfaces, porous solids) by defining and controlling the relief (surface state) and lacunarity (size and distribution of holes). In this context we intend to develop differential calculus tools for fractal curves and surfaces defined by IFS. Using local fractional derivatives, we show that, even if most fractal curves are nowhere differentiable, they admit a left and right half-tangents, what gives us an additional parameter to characterize shapes.

Keywords: fractal curve, fractal surface, iterated function systems, differentiability.

1 Introduction

Our long-term goal is to develop a geometric modeler based on iterative process and fractal geometry to allow designers to access a new universe of shapes. Special properties of fractal structures have led us to new concepts inexistent for classical geometric objects. Fractal curves and surfaces, for example, can have very different aspects and very different kinds of roughness. This vast variety of shapes is not accessible with polynomial curves and surfaces precisely because of their differentiability.

To control these aspects we are led to use the concept of "geometric texture" [2]. This "geometric texture" is very tightly coupled with differentiability of curves and surfaces. We study and attempt to characterize differential behavior from a geometric point of view by means of local fractional derivative [3].

Of course, other works were performed to study differential properties of fractal curves and surfaces. Kolwankar and Gangal [12,11] applied the fractional calculus to study real-valued functions with few examples of fractal curves. Using their results the authors are able to describe roughness of a curve with the Hölder exponent.

Cochran has proposed a method of calculating normals to a fractal surface [6]. Scealy [16] identifies C^1 fractal curves. However, these works are applicable in the cases where derivatives exist and do not permit to characterize rough shapes.

In this paper we present a general approach to study differential properties of fractal curves and surfaces. We are using BC-IFS (Boundary Controled Iterated Function System) [19] to construct fractal structures. Then we are using this representation to study necessary and sufficient conditions of differentiability.

In order to simplify the presentation we show in detail how it can be done for the family of local corner cutting curves. Differential behaviour of these curves was studied before by De Boor [5], Gregory [8] and Paluszny [14], the cited authors have found necessary conditions of differentiability. However, BC-IFS approach allows to describe a larger family of curves, and therefore while we find the same necessary conditions for the set of curves given by De Boor, we also study other regions of the convergence domain. A cartography of domains is presented in late sections of the paper. Finally, we show that necessary conditions are also sufficient ones.

The rest of the paper is organized as follows:

– Section 2 provides necessary background needed to introduce BC-IFS in Section 3.
– Section 4 studies necessary conditions of differentiability and presents cartography of the convergence domain
– Section 5 shows that necessary conditions are also sufficient ones
– Section 6 introduces a new descriptor of roughness of a fractal shape.

2 Background

2.1 IFS

Given a complete metric space (E, d), where d is the associated metric, an *IFS* (Iterated Function System) is a finite set of contractive operators $\mathbb{T} = \{T_i\}_{i=0}^{N-1}$ acting on points of E. Each $T_i : E \to E$ induces $T_i : \mathcal{H}(E) \to \mathcal{H}(E)$, i.e. operators acting in the space $\mathcal{H}(E)$ of non-empty compact subsets of E.

Thus it is possible to define so-called Hutchinson operator $\mathbb{T} : \mathcal{H}(E) \to \mathcal{H}(E)$ as a union of operators T_i. The Hutchinson operator maps a non-empty compact $K \subset E$ onto $\bigcup\limits_{i=0}^{N-1} T_i(K)$. The operators T_i are contractive in the space (E, d), therefore the induced operators are contracting in the space $(\mathcal{H}(E), d_{\mathcal{H}(E)})$, where $d_{\mathcal{H}(E)}$ is the Hausdorff metric [1]. Of course, the Hutchinson operator is also contractive in $(\mathcal{H}(E), d_{\mathcal{H}(E)})$.

The contraction theorem [9] states that there is a unique compact A such as $\mathbb{T}(A) = A$, namely the fixed point, noted $\mathcal{A}(\mathbb{T})$. Moreover, the fixed point A may be found as a limit $A = \lim\limits_{n\to\infty} \mathbb{T}^n(K)$, where the limit does not depend on the choice of the "seed" compact K. The top line of Figure 2 provides an illustration. The underlying IFS is composed of four transformations:

$$T_0\begin{pmatrix}x\\y\end{pmatrix} = \begin{bmatrix}0.85 & 0.04\\-0.04 & 0.85\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix} + \begin{bmatrix}0.00\\1.60\end{bmatrix} \qquad T_1\begin{pmatrix}x\\y\end{pmatrix} = \begin{bmatrix}-0.15 & 0.28\\0.26 & 0.24\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix} + \begin{bmatrix}0.00\\0.44\end{bmatrix}$$

$$T_2\begin{pmatrix}x\\y\end{pmatrix} = \begin{bmatrix}0.20 & -0.26\\0.23 & 0.22\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix} + \begin{bmatrix}0.00\\1.60\end{bmatrix} \qquad T_3\begin{pmatrix}x\\y\end{pmatrix} = \begin{bmatrix}0.00 & 0.00\\0.00 & 0.16\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix}$$

We have chosen a square as the seed K. Remember that the final shape is independent of the choice. Thus, at the first iteration we apply each $\{T_i\}_{i=0}^{n-1}$ to the square to get four deformed quadrilaterals in place of two branches, the stem and the top of the fern. Then we take a union of the quadrilaterals and restart the process. In few iterations only, quadrilaterals vanish being almost imperceptible, but their union being plenty engender the shape of the fern.

2.2 CIFS

In regular IFS we start from a seed, then apply a set of rules (transformations), and repeat as required. In CIFS (Controlled, or graph-directed IFS) not all rules need to be applied at each step, a directed graph controls (directs) rules [13,15]. We associate work spaces to the nodes of the graph and the arcs represent the transformations to be applied at the current state. In such a way it is possible to blend attractors of different nature.

The left image of Figure 1 represents the control graph (it can be seen as an automaton) for the regular IFS generating the Barnsley fern, the iteration process is shown in the top line of Figure 2. But what happens if we modify the automaton? Let us add three more transformations to the IFS (the corresponding automaton is shown on the right of Figure 1):

$$T_5\begin{pmatrix}x\\y\end{pmatrix} = \begin{bmatrix}1/2 & 0\\0 & 1/2\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix} \qquad T_6\begin{pmatrix}x\\y\end{pmatrix} = \begin{bmatrix}1/2 & 0\\0 & 1/2\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix} + \begin{bmatrix}1/2\\0\end{bmatrix}$$

$$T_7\begin{pmatrix}x\\y\end{pmatrix} = \begin{bmatrix}1/2 & 0\\0 & 1/2\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix} + \begin{bmatrix}0\\1/2\end{bmatrix}$$

These three transformations on itself generate the Sierpiński's triangle. Note also that the destination of the transformation T_2 is changed. Thus, once the transformation T_2 was applied, the subdivision is made according to the rules of the Sierpiński's triangle. While the Barnsley's fern consists of infinite number of shrunk copies of itself, the attractor shown in the right bottom image of Figure 2 is a fern that consists of infinite number of shrunk Sierpiński's triangles[1].

[1] In fact, the arrows of the automaton depict the data flow, or the order of application of transformations. However actual transformations act in the other direction. That is so, the right lowest branch of the new fern can be found as the following limit: $T_2(\mathcal{A}(\mathbb{T}_{5,6,7}))$, where $\mathbb{T}_{5,6,7}(K) = T_5(K) \cup T_6(K) \cup T_7(K)$. This implies that we have to choose two seeding compacts for two different spaces, in the images we have chosen a square and a triangle.

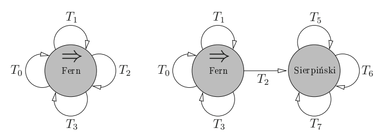


Fig. 1. Two automatons generating rules (order) of application of transformations

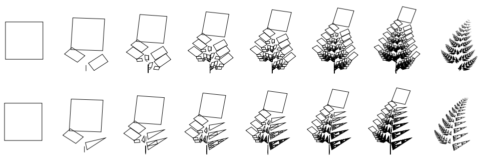


Fig. 2. Top line: the Barnsley fern; bottom line : C-IFS allows to mix up attractors of different nature

2.3 Projected IFS

The notion of projected IFS was introduced by Zaïr and Tosan [20]. If one separates the iteration space from the modelling space, it is possible to create free-form fractal shapes. The work was inspired by spline curves, which are created by a projection of basis functions defined in a barycentric space. In the same way, it is possible to construct an IFS attractor in a barycentric space (whose dimension is equal to the number of control points) and to project it into the modelling space. In other words, if we have an attractor $A \subset BI^n = \{\lambda \in \mathbb{R}^n | \sum_{i=0}^{n-1} \lambda_i = 1\}$, where n is the number of control points, the projection can be made just by a matrix multiplication $PA = \{\sum_{i=0}^{n-1} P_i\lambda_i | \lambda_i \in A\}$. Here the matrix $P = [P_0 \ P_1 \cdots P_{n-1}]$ is composed of control points. This construction imposes that transformations in IFS must act in a barycentric space. For linear operators expressed in matrix form it means that all columns sum up to 1.

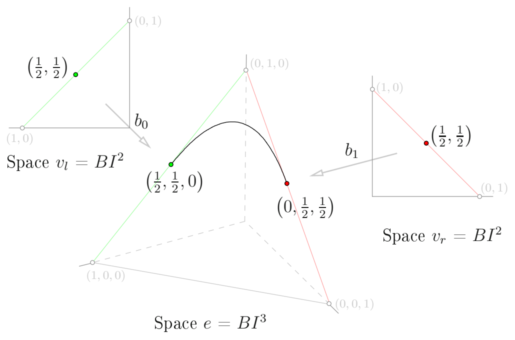
3 Boundary Controlled IFS

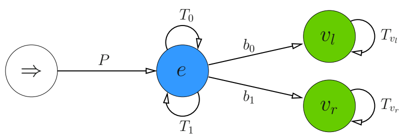
Boundary Controlled IFS (BC-IFS) is a graph-controlled IFS with a B-rep structure introduced by Tosan et al [19]. This is a convenient method to express face-edge-vertex hierarchies implicitly existing in many fractal attractors [7]. The notions of B-rep here are a bit more general that in the classical case. Here a topological cell may be bordered by a fractal object and not only with an edge (vertex). For example, a "face" may be the Sierpiński's triangle, an "edge" the Cantor set. The advantage of this method is its power to express incidence and adjacency constraints for subdivision processes for a given topology, what results into constraints in the subdivision matrices.

To define free-form shapes with BC-IFS it is necessary to distinguish different work spaces:

– the modeling space is where the final shape lives, this is also the space where we place control points;
– barycentric spaces where we construct attractors corresponding to different topological entities, and this is where IFS transformations act.

Let us illustrate the approach by constructing a local corner cutting 2D or 3D curve. This type of curves demands at least 3 control points, and endpoints of


(0, 1)
$(\frac{1}{2}, \frac{1}{2})$
b_0
(1, 0)
Space $v_l = BI^2$
(0, 1, 0)
$(\frac{1}{2}, \frac{1}{2}, 0)$
$(0, \frac{1}{2}, \frac{1}{2})$
(1, 0, 0)
(0, 0, 1)
Space $e = BI^3$
b_1
(1, 0)
$(\frac{1}{2}, \frac{1}{2})$
(0, 1)
Space $v_r = BI^2$



$$\begin{tikzcd}
\Rightarrow \arrow[r, "P"] & e \arrow[loop above, "T_0"] \arrow[loop below, "T_1"] \arrow[r, "b_0"] \arrow[dr, "b_1"'] & v_l \arrow[loop right, "T_{v_l}"] \\
& & v_r \arrow[loop right, "T_{v_r}"]
\end{tikzcd}$$

Fig. 3. At the left and at the right : barycentric spaces $v_l = v_r = BI^2$ corresponding to the vertices (left and right respectively); in the middle: edge barycentric space $e = BI^3$. The operators b_0 and b_1 embed the spaces v_l and v_r into subspaces of e.

Fig. 4. General edge-vertex B-rep BC-IFS automaton

a curve depend on two of them. For a curve the B-rep structure is simple: we will have edges bounded by vertices, therefore, in general case we will have four different spaces:

– $\mathbb{R}^2$ or $\mathbb{R}^3$ where the final curve is to be drawn, the control points are to be placed here
– a barycentric space of dimension 3 (we construct the simplest case with three control points), this space is where the attractor of the B-rep "edge" lives
– a barycentric space of "left" endpoint of the edge, the dimension is 2 since it depends on two control points
– similary a two-dimensional barycentric space for the "right" endpoint.

The edge and vertices are attractors in barycentric space to be projected to the modeling space, thus we need three IFS to build the attractors. Let us say that the edge is obtained with an IFS $\{T_0, T_1\}$, where T_0 and T_1 are 3×3 subdivision matrices for the edge. The vertices are obtained with IFS $\{T_{v_l}\}$ and $\{T_{v_r}\}$, and the matrices are 2×2. Figure 4 shows the BC-IFS automaton. First of all we see four nodes corresponding to four spaces. The matrix P is the projection matrix composed of control points. The only thing we have not yet defined are transformations b_0 and b_1.

In fact, up to this moment we have not imposed any constraints on the IFS matrices. If we fill them with random coefficients, nothing guarantees any connectivity. However, in B-rep, vertices are boundaries of the edge, so there must be some relationship between the matrices. To ensure this we need embedding operators, namely b_0 and b_1.

Figure 3 illustrates the approach. It shows the basis functions of a uniform B-spline quadratic curve drawn in the three-dimensional barycentric space

$e = BI^3$. Basis functions for endpoints (in fact, these are just points) of the curve are drawn in corresponding two-dimensional spaces $v_l = BI^2$ and $v_r = BI^2$. Then b_0 and b_1 embed endpoint spaces into the edge space to impose that the edge has the vertices for its endpoints. Let us find shapes of b_0 and b_1. The endpoints have coordinates $\begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}$ in the spaces v_l and v_r. At the same time in the space e they are $\begin{pmatrix} 1/2 \\ 1/2 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1/2 \\ 1/2 \end{pmatrix}$. Therefore, the mappings $b_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$ and $b_1 = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$ are indeed simple embeddings. In other words, b_0 and b_1 say on which control points depend corresponding endpoints.

3.1 Topology Constraints

Incidence and adjacency constraints may be easily obtained by expanding the control graph. Figure 5 is an unfolded version of Figure 4 which describes the subdivision process of our example. After the first iteration on the control graph we pass from the modeling space to the edge space e, where an edge is bounded by two vertices v_l and v_r. This situation is shown in the top line of the figure. After one more iteration (bottom line) the edge is subdivided by T_0 and T_1 in two smaller edges along with their own two endpoints.

Adjacency Constraints. Let us say that we want to get C_0 continuity. When an edge is split into two edges, the "left" subdivided edge must be connected to the "right" one in order to guarantee the topology of a curve. Therefore, we impose the "right" endpoint of the "left" edge to coincide with the "left" endpoint of the "right" edge and this implies that the "left" and "right" endpoints are of the same nature and actually live in the same space having common generating IFS. Otherwise, the connectivity will be broken at following stages of the subdivision process. So we have $T_{v_l} = T_{v_r} = T_v$.

When we say the "right" endpoint of the "left" edge this means that we can follow the path $e \xrightarrow{T_0} e \xrightarrow{b_1} v$ in the control graph. The same holds for the "left" endpoint of the "right" edge : $e \xrightarrow{T_1} e \xrightarrow{b_0} v$. As mentioned above, the vertices coincide, thus we can write $T_0 b_1 = T_1 b_0$.

Let us fill T_0 and T_1 with some arbitrary coefficients:

$$T_0 = \begin{pmatrix} a_0 & b_0 & c_0 \\ d_0 & e_0 & f_0 \\ g_0 & h_0 & i_0 \end{pmatrix}, \qquad T_1 = \begin{pmatrix} a_1 & b_1 & c_1 \\ d_1 & e_1 & f_1 \\ g_1 & h_1 & i_1 \end{pmatrix}.$$

Then we rewrite the constraint: $T_0 b_1 = T_1 b_0$

$$\begin{pmatrix} a_0 & b_0 & c_0 \\ d_0 & e_0 & f_0 \\ g_0 & h_0 & i_0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} a_1 & b_1 & c_1 \\ d_1 & e_1 & f_1 \\ g_1 & h_1 & i_1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}.$$

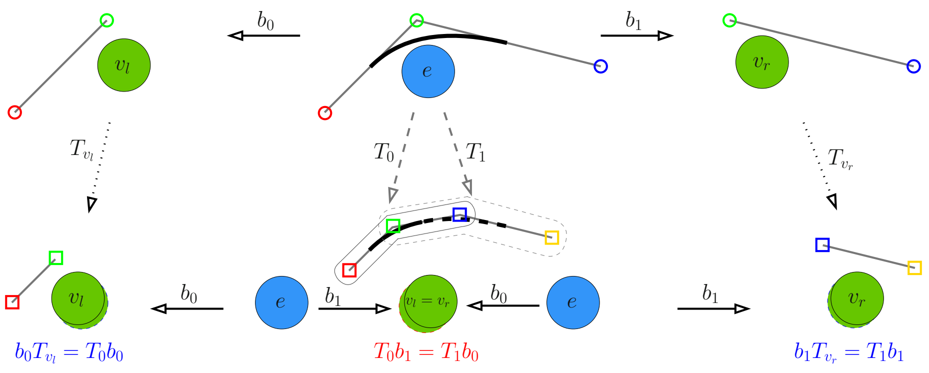


Fig. 5. Unfolded version of the control graph. This subdivision must be constrained in order to get the desired topology (here a curve). Incidence constraints are at the left and right, while adjacency constraint is in the middle.

This implies that the last two columns of T_0 are equal to the first two columns of T_1:

$$\begin{pmatrix} b_0 & c_0 \\ e_0 & f_0 \\ h_0 & i_0 \end{pmatrix} = \begin{pmatrix} a_1 & b_1 \\ d_1 & e_1 \\ g_1 & h_1 \end{pmatrix} .$$

Incidence Constraints. In the same manner, incidence constraints may be deduced from the fact that the subdivision of the endpoints must be in harmony with endpoints of subdivided edges. Thus, for the left endpoint let us follow the paths $e \xrightarrow{b_0} e \xrightarrow{T_v} v = e \xrightarrow{T_0} e \xrightarrow{b_0} v$, what results into the constraint $b_0 T_v = T_0 b_0$. The same holds for the right endpoint: $b_1 T_v = T_1 b_1$, the constraints are shown in Figure 5. If the constraints are not fulfilled we will get a disconnected curve after two subdivisions. Let us solve the constraints on the matrices. Having denoted $T_v = \begin{pmatrix} a_v & b_v \\ d_v & e_v \end{pmatrix}$ we get

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} a_v & b_v \\ d_v & e_v \end{pmatrix} = \begin{pmatrix} a_0 & b_0 & c_0 \\ d_0 & e_0 & f_0 \\ g_0 & h_0 & i_0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} a_v & b_v \\ d_v & e_v \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} a_0 & b_0 \\ d_0 & e_0 \\ g_0 & h_0 \end{pmatrix}$$

and

$$\begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} a_v & b_v \\ d_v & e_v \end{pmatrix} = \begin{pmatrix} a_1 & b_1 & c_1 \\ d_1 & e_1 & f_1 \\ g_1 & h_1 & i_1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 0 \\ a_v & b_v \\ d_v & e_v \end{pmatrix} = \begin{pmatrix} b_1 & c_1 \\ e_1 & f_1 \\ h_1 & i_1 \end{pmatrix}$$

Then it easy to see that

$$T_0 = \begin{pmatrix} a_s & b_s & 0 \\ d_s & e_s & a_s \\ 0 & 0 & e_s \end{pmatrix} \qquad T_1 = \begin{pmatrix} b_s & 0 & 0 \\ e_s & a_s & b_s \\ 0 & d_s & e_s \end{pmatrix}.$$

Let us add the fact that the matrices are stochastic (all columns sum up to 1) and we see that for a local corner cutting curve whose points depend on at most three control points (and on two at least) there are only two degrees of freedom:

$$T_0 = \begin{bmatrix} 1-a & b & 0 \\ a & 1-b & 1-a \\ 0 & 0 & a \end{bmatrix} \qquad T_1 = \begin{bmatrix} b & 0 & 0 \\ 1-b & 1-a & b \\ 0 & a & 1-b \end{bmatrix}$$

Convergence. The convergence theorem [1] states that in order to get the convergence of an IFS with linear transformations the operators must have eigenvalues strictly less than 1 (in absolute value) with one exception: all stochastic operators have eigenvalue 1 that correspond to fixed points of the operators. T_0 has eigenvalues: $(1, a, 1-a-b)$, while T_1 has $(1, b, 1-a-b)$. Thus in our case the convergence holds if and only if

$$\begin{aligned} -1 &< a < 1 \\ -1 &< b < 1 \\ -1 &< 1-a-b < 1 \end{aligned}$$

Local corner cutting curves were studied earlier by Gregory, Qu, De Boor et al [8,5,14]. The notations we use here correspond exactly to their works, however there is a difference in the domain of definition. In fact, when the cited authors construct corner cutting curves, they suppose that all vertices of a polygon at iteration n belong to the polygon from the iteration $n-1$. Therefore, the studied domain is shown in horizontal hatching in Figure 6, it corresponds to the domain with positive eigenvalues a, b and $1-a-b$. Our construction does not use this assumption, so the domain we study here is all the region of convergence (shown in vertical hatching).

Parameterization and Self-similarity. Under latter constraints the attractor of the IFS $\{T_0, T_1\}$ is a curve in three-dimensional barycentric space. It is easy to parameterize the curve with so-called natural parameterization $t \in [0,1]$, where $t = 0$ corresponds to the "left" endpoint of the curve, $t = 1$ is the "right" endpoint and $t = \frac{1}{2}$ corresponds to the junction point in the first level of subdivision.

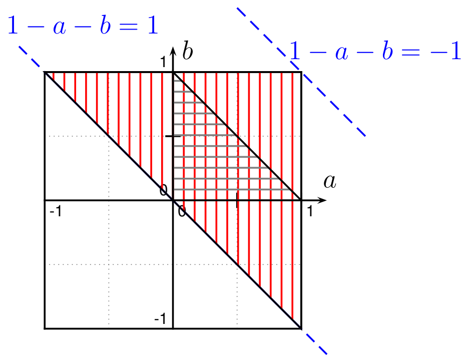


Fig. 6. In horizontal hatching: the domain of convergence, in gray: Gregory-Qu domain of definition

Then if we denote the parameterized curve (in the barycentric space) as $F(t)$, it is easy to get the parameterized curve in the modelling space $C(t) = PF(t)$, where $P = \begin{pmatrix} P_0 & P_1 & P_2 \end{pmatrix}$ is the vector of control points. Let us rewrite the self-similarity property of the curve: $F\left(\left[0, \frac{1}{2}\right]\right) = T_0 F([0,1])$ and $F\left(\left[\frac{1}{2}, 1\right]\right) = T_1 F([0,1])$. The parameterization is induced by the subdivision of the parameter space $[0,1]$ for each iteration.

4 Differentiability

Half-tangent vectors at endpoints are defined for large class of fractal curves [3]. Attractors are self-similar, so if a half-tangent $\overrightarrow{v}$ exists for $t = 0$ then it is easy to find a half-tangent vector for $t = \frac{1}{2}$: vector $T_1 \overrightarrow{v}$ is tangent to the curve $F\left(\left[\frac{1}{2}, 1\right]\right) = T_1 F([0,1])$. Therefore, having defined half-tangent vectors for endpoints of a fractal curve we automatically define it for a set dense in the parameter domain. In the same way if a fractal curve is not differentiable for an endpoint the singularity is copied by the self-similarity property.

4.1 Eigenvectors and Eigenvalues

T_0 has real eigenvalues $\lambda_0^0 = 1$, $\lambda_1^0 = 1 - a - b$, $\lambda_2^0 = a$ and T_1 has $\lambda_0^1 = 1$, $\lambda_1^1 = 1 - a - b$, $\lambda_2^1 = b$. Corresponding eigenvectors are:

$$\overrightarrow{v_0^0} = \begin{pmatrix} \frac{b}{a+b} \\ \frac{a}{a+b} \\ 0 \end{pmatrix}, \overrightarrow{v_1^0} = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}, \overrightarrow{v_2^0} = \begin{pmatrix} -b \\ 1-2a \\ 2a+b-1 \end{pmatrix}$$

and

$$\overrightarrow{v_0^1} = \begin{pmatrix} 0 \\ \frac{b}{a+b} \\ \frac{a}{a+b} \end{pmatrix}, \overrightarrow{v_1^1} = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}, \overrightarrow{v_2^1} = \begin{pmatrix} a+2b-1 \\ 1-2b \\ -a \end{pmatrix}.$$

Eigenvectors v_0^0 and v_0^1 corresponding to the eigenvalue 1 give fixed points of T_0 and T_1. Note that the fixed points are endpoints of the curve: $v_0^0 = F(0)$ and $v_0^1 = F(1)$. It is easy to see that eigenvectors corresponding to the sub-dominant eigenvalues give half-tangents to the endpoints. Note that sub-dominant eigenvalues of T_0 and T_1 are non negative.

There are four cases:

1. $1-a-b$ is the sub-dominant eigenvalue of T_0: $|1-a-b| \geq |a| \Rightarrow 1-2a-b \geq 0$. In such a case the half-tangent to $F(0)$ is collinear with $\begin{pmatrix}1 & -1 & 0\end{pmatrix}^{\mathsf{T}}$ and the half-tangent to $C(0)$ is collinear with $\overrightarrow{P_0 - P_1}$.
2. a is the sub-dominant eigenvalue of T_0: $|a| > |1-a-b| \Rightarrow 1-2a-b < 0$. In this case the half-tangent to $C(0)$ is $-bP_0 + (1-2a)P_1 + (2a+b-1)P_2$. This vector can have different orientations depending on the values of a and b.
3. $1-a-b$ is the sub-dominant eigenvalue of T_1: $|1-a-b| \geq |b| \Rightarrow 1-a-2b \geq 0$. The half-tangent to $C(1)$ is collinear with $\overrightarrow{P_1 - P_2}$.
4. b is the sub-dominant eigenvalue of T_1: $|b| > |1-a-b| \Rightarrow 1-a-2b < 0$. The half-tangent to $C(1)$ has direction $(a+2b-1)P_0 + (1-2b)P_1 - aP_2$. Again, the direction depends on a and b.

The most interesting case is when $1-a-b$ is the sub-dominant eigenvalue for both T_0 and T_1. In such a case half-tangents are given by two control points and therefore their directions do not depend on a and b.

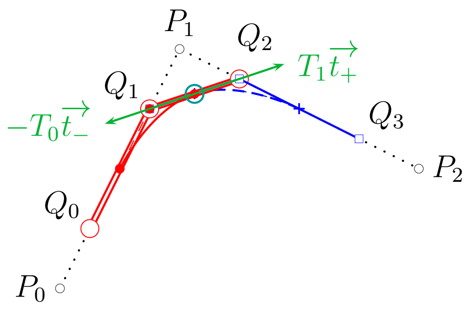


Fig. 7. Half-tangent vectors at the joining point

4.2 Necessary Conditions for Differentiability

Incidence and adjacency constraints of the BC-IFS guarantee C^0 continuity for the limit curve. To have C^1 continuity half-tangents must be collinear at the junction point. Figure 7 shows an illustration.

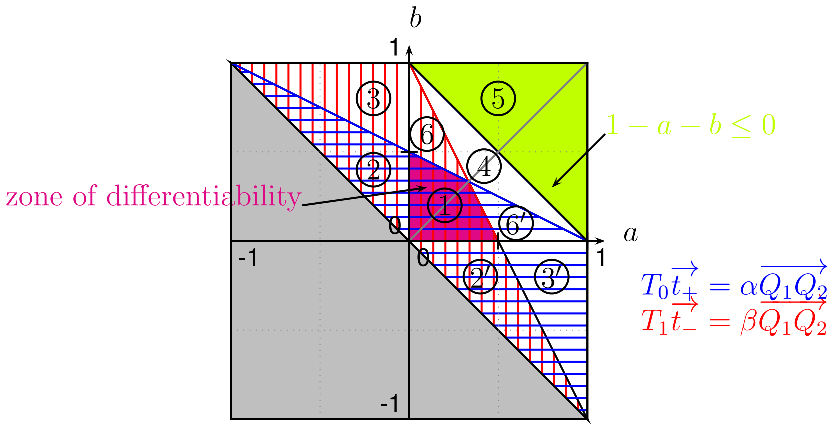
Let us suppose that the curve is differentiable, the half-tangents for the point $t = \frac{1}{2}$ may be obtained by the self-similarity property from the half-tangents to endpoints. If $\overrightarrow{t_-}$ and $\overrightarrow{t_+}$ are the directions of the half-tangents to endpoints, then $T_1\overrightarrow{t_+}$ must be equal to $T_0\overrightarrow{t_-}$:

$$\begin{aligned} C([0,1]) &= \big(Q_0\ Q_1\ Q_2\big)\, F([0,1]) \quad \cup \quad \big(Q_1\ Q_2\ Q_3\big)\, F([0,1]) \\ &= PT_0F([0,1]) \quad \cup \quad PT_1F([0,1]) \end{aligned}$$

As we have mentioned previously, vectors $\overrightarrow{t_+}$ and $\overrightarrow{t_-}$ depend on values of a and b. To have G^1 continuity, it is obvious that $T_1\overrightarrow{t_+}$ and $T_0\overrightarrow{t_-}$ must be (at least) collinear for any configuration of control points $\big(Q_0\ Q_1\ Q_2\ Q_3\big)$. The only possibility to fulfil the collinearity is when[2]:

- vector $\overrightarrow{t_-}$ belongs to the subspace corresponding to the two last control points P_1 and P_2, i.e. has zero first component. This is the case iff $1-2a-b>0$.
- vector $\overrightarrow{t_+}$ belongs to the subspace corresponding to the two first control points P_0 and P_1, i.e. has zero third component. This is the case iff $1-a-2b>0$.

Therefore if $T_1\overrightarrow{t_-}$ and $T_0\overrightarrow{t_+}$ are collinear, then $1-a-b$ is sub-dominant eigenvalue for both T_0 and T_1. The corresponding domain is shown by horizontal and vertical hatching in Figure 8. However it is a *necessary* condition: it includes regions of differentiability (zone 1) as well as regions of cusp points (zones 2 and 2′). Therefore, collinearity is a rough tool and to distinguish the zones we have to find direction of tangent vectors.



b
1
a
1
-1
-1
0
0
5
3
6
4
2
1
6′
2′
3′
$1-a-b\le 0$
zone of differentiability
$T_0\overrightarrow{t_+} = \alpha\overrightarrow{Q_1Q_2}$
$T_1\overrightarrow{t_-} = \beta\overrightarrow{Q_1Q_2}$

Fig. 8. Cartography of regions according to differential properties

4.3 Cartography of Differential Behaviours

To find direction of half-tangents and to identify differential behaviour of the other areas in the convergence domain, we use the following property etablished in [2].

[2] Except degenerated cases: if $a=0$ or $b=0$ or $a+b=1$ then the curve is degenerated (piecewise linear) and therefore differentiable (except at vertices).

Property 1. Let us find decomposition of $\overrightarrow{F(0)F(1)}$ in the eigenbases of T_0 and T_1, respectively:

$$\overrightarrow{F(0)F(1)} = \alpha_1\overrightarrow{v_1^0} + \alpha_2\overrightarrow{v_2^0} = \frac{(-b)(a+b-1)}{(2a+b-1)(a+b)}\overrightarrow{v_1^0} + \frac{a}{(2a+b-1)(a+b)}\overrightarrow{v_2^0}$$
$$\overrightarrow{F(0)F(1)} = \beta_1\overrightarrow{v_1^1} + \beta_2\overrightarrow{v_2^1} = \frac{(-a)(a+b-1)}{(a+2b-1)(a+b)}\overrightarrow{v_1^1} + \frac{-b}{(a+2b-1)(a+b)}\overrightarrow{v_2^1}$$

Let $R, L \in \{1,2\}$ such that v_L^0 and v_R^1 are the sub-dominant eigenvectors of T^0 and T^1, respectively. Then if $F(t)$ has left and right half-tangents at respectively $F(1)$ and $F(0)$, their directions $\overrightarrow{t_-}$ and $\overrightarrow{t_+}$ are given by:

$$\overrightarrow{t_-} = \alpha_L\overrightarrow{v_L^0}$$
$$\overrightarrow{t_+} = \beta_R\overrightarrow{v_R^1}.$$

Let us consider the subdivision of the curve :

$$C([0,1]) = \begin{pmatrix} Q_0 & Q_1 & Q_2 \end{pmatrix} F([0,1]) \cup \begin{pmatrix} Q_1 & Q_2 & Q_3 \end{pmatrix} F([0,1])$$

Now we focus on the point of junction of the two sub-curves $C_0(t) = \begin{pmatrix} Q_0 & Q_1 & Q_2 \end{pmatrix} F(t)$ and $C_1(t) = \begin{pmatrix} Q_1 & Q_2 & Q_3 \end{pmatrix} F(t)$. The directions of the half-tangents at the point are given by $\begin{pmatrix} Q_1 & Q_2 & Q_3 \end{pmatrix} \overrightarrow{t_-}$ for C_0 and $\begin{pmatrix} Q_0 & Q_1 & Q_2 \end{pmatrix} \overrightarrow{t_+}$ for C_1.

Depending on sub-dominant eigenvalues of T_0 and T_1 we can have three main different cases:

1. $1-a-b > a$ and $1-a-b > b$: this case covers three regions in Figure 8, namely regions 1, 2 and 2'. Here we have $R = L = 1$ and

$$\begin{pmatrix} Q_0 & Q_1 & Q_2 \end{pmatrix} \overrightarrow{t_r} = \beta_1 \begin{pmatrix} Q_0 & Q_1 & Q_2 \end{pmatrix} \overrightarrow{v_1^1} = \beta_1\overrightarrow{Q_1Q_2}$$
$$\begin{pmatrix} Q_1 & Q_2 & Q_3 \end{pmatrix} \overrightarrow{t_l} = \alpha_1 \begin{pmatrix} Q_1 & Q_2 & Q_3 \end{pmatrix} \overrightarrow{v_1^0} = \alpha_1\overrightarrow{Q_1Q_2}$$

   with $\alpha_1 = \frac{(-b)(a+b-1)}{(2a+b-1)(a+b)}$ and $\beta_1 = \frac{(-a)(a+b-1)}{(a+2b-1)(a+b)}$. As was explained in the previous section, the two half-tangent vectors at the joining point are collinear in this case. However, the vectors have the same direction if and only if α_1 and β_1 are of the same sign, and it is the case for the region 1 of Figure 8. For regions 2 and $2'$ half-tangent vectors have opposite directions, resulting into cusp points.
2. $1-a-b > a$ or (exclusive) $1-a-b > b$: $\begin{pmatrix} Q_1 & Q_2 & Q_3 \end{pmatrix} \overrightarrow{t_+}$ and $\begin{pmatrix} Q_0 & Q_1 & Q_2 \end{pmatrix} \overrightarrow{t_-}$ are not collinear in general case. If one of the sub-dominant eigenvalues of T_0 or T_1 is $1-a-b$ then the corresponding half-tangent vector is colinear with $\overrightarrow{Q_1Q_2}$ (regions 3 and $3'$) but the other half-tangent is not.
3. $1-a-b < a$ and $1-a-b < b$: this case corresponds to regions 4 and 5 of Figure 8. No half-tangent vector is collinear to $\overrightarrow{Q_1Q_2}$ since the eigenvectors have three non-zero components and therefore the half-tangent vectors

depend on three control points respectively $(Q_1\, Q_2\, Q_3)$ and $(Q_0\, Q_1\, Q_2)$. Regions 4 and 5 differ in the sign of the smallest (in absolute value) eigenvalue $1 - a - b$. For the region 5 the eigenvalue is negative, and it forces the curve to oscillate around the direction of the half-tangent, thus giving a "fractal" aspect to the curve.

5 Sufficient Conditions

Figure 9 shows the motivation for this section. If we use the natural parameterization, then even for differentiable curves, blending functions $F(t)$ are not differentiable in the sense of Lipschitz. However, under a suitable parameterization the blending functions are differentiable. The image is obtained for values $a = 1/20$ and $b = 1/8$.

This is very similar to the situation with Stam's method [18] of exact evaluation of subdivision surfaces. Having constructed the natural parameterization it is easy to find points of a curve (surface), however the behaivour of derivatives is erratic and therefore many methods like [10] may fail to work with this parameterization. There are several works that construct non-singular parameterizations, for example, we can cite [4] for Catmull-Clark subdivision surfaces.

In this section we will show how to reparameterize any curve from Gregory region to garantee C^1 blending functions. So instead of subdividing the parameter domain in equal halves as the natural parameterization does, we follow the same subdivision rules as for the control polygon. Figure 10 illustrates the idea. We start with a control polygon (P_0, P_1, P_2); in order to parameterize it we chose three real values (t_0, t_1, t_2) such that $t_0 < t_1 < t_2$. Then we say that the segments (P_0, P_1) and (P_1, P_2) have linear parameter domains (t_0, t_1) and (t_1, t_2), respectively.

Then subdivided polygon (Q_0, Q_1, Q_2, Q_3) is parameterized with three segments (u_0, u_1), (u_1, u_2) and (u_2, u_3), where u_i are obtained by the same rules of subdivision as Q_i:

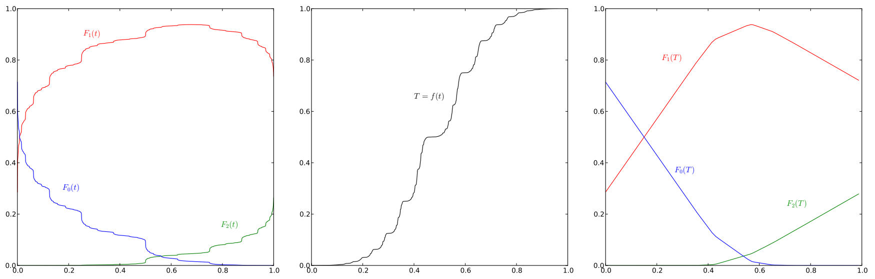


Fig. 9. Left image: The three components of blending function $F(t)$ in the natural parameterization. In the middle: non-singular parameterization as a functtion of natural parametrization. Right image: The three components blending function in the non-singular parameterization.

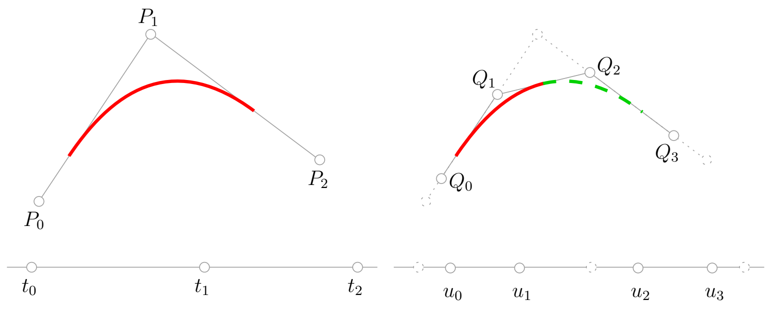


Fig. 10. Left image: original control polygon (P_0, P_1, P_2) and its parameter values (t_0, t_1, t_2). Right image: parameter domain is subdivided along with the control polygon.

$$\begin{aligned} \left(Q_0\; Q_1\; Q_2\right) &= \left(P_0\; P_1\; P_2\right) T_0 \qquad & \left(u_0\; u_1\; u_2\right) &= \left(t_0\; t_1\; t_2\right) T_0 \\ \left(Q_1\; Q_2\; Q_3\right) &= \left(P_0\; P_1\; P_2\right) T_1 \qquad & \left(u_1\; u_2\; u_3\right) &= \left(t_0\; t_1\; t_2\right) T_1. \end{aligned}$$

The limit of the process gives us a well-parameterized curve. To verify the C^1 continuity of the limit curve one may proceed as follows:

- construct a sequence of functions $\{f_i\}_{i=0}^{\infty}$ converging pointwise to the limit function $F(t)$. Here we start with a vector of blending functions f_0 for the control polygon (P_0, P_1, P_2). Then f_1 is the vector of blending functions for the polygon (Q_0, Q_1, Q_2, Q_3) etc.
- construct a sequence of derivatives $\{f_i'\}_{i=0}^{\infty}$ and show that it converges uniformly to a continuous function
- prove that the limit $\lim\limits_{i\to\infty} f_i'$ is indeed the derivative of the curve $F(t)$.

In such a way we get *sufficient* conditions for differentiability, not only necessary ones. We do not want to overload the presentation with technical questions of uniform convergence, all the proofs are detailed in a technical report [17]. The report proves that a curves is C^1 continuous if and only if a and b are located in the Gregory region (magenta zone in Figure 8).

Moreover, we have proved that for *any* a and b in the convergence domain limit curves are differentiable *almost everywhere*, i.e. everywhere except on a set of measure zero [17]. As a matter of fact, this set consists of the junction point under all possible finite sequences of applications T_0 and T_1. In other words, in the natural parameterization it is the point $t = \frac{1}{2}$, $t = \frac{1}{4}$, $t = \frac{3}{4}$ etc (all points of dyadic parameters). This set is denumerable.

6 Roughness of a Curve

There are few ways to describe roughness of a curve like Hölder exponent and fractal dimension. All the descriptors are good per se, but a curve may be fully

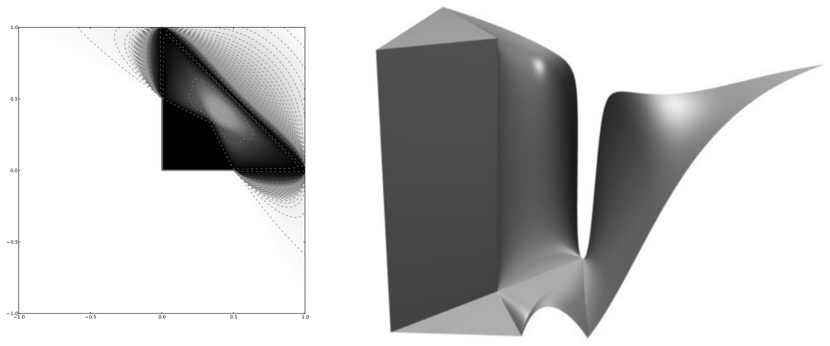


Fig. 11. Left image: angle between two half-tangent vectors from 0 (black) to π (white); right image: the same, but represented as a 3d surface (cut along the symmetry line $x = y$ to show the interior)

described only by combining descriptors. Here we introduce a new descriptor, namely angles between half-tangent vectors $T_0\overrightarrow{t_-}$ and $T_1\overrightarrow{t_+}$.

So we know that any curve from the Gregory-Qu domain is differentiable, but if we are not very far from the domain; curves are not very rough either. These "almost smooth" curves may be a good fit for computer graphics, where all geometry is discretized anyway. Or if one searches for a really rough curve, where to look for it in the convergence domain? Angles between half-tangent vectors are very easy to calculate, and therefore the search is very efficient:

$$\alpha(a, b) = \arccos \frac{< T_0\overrightarrow{t_-}, T_1\overrightarrow{t_+} >}{\|T_0\overrightarrow{t_-}\| \|T_1\overrightarrow{t_+}\|}$$

Left image of Figure 11 shows a graph of roughness vs values of a and b. The domain of Gregory-Qu is marked in black as half-tangent are collinear (note that degenerate cases $a = b$, $b = 0$ and $1 = a + b$ are also in black). All cusp points are marked in white.

7 Conclusion

So far we have shown how a curve may be constructed. Constructing a surface may be done in exactly the same manner. For example, Doo-Sabin subdivision scheme may be described as a face-edge-vertex B-rep, where a patch (topological "face") is bounded by four "edges". Then each patch is subdivided into four smaller sub-patches and all them may be stitched together by implying adjacency of corresponding borders. Then it is immediate that for a (regular) Doo-Sabin patch there are three degrees of freedom. Either we set it to classic values $(0.5625, 0.1875, 0.1875)$ to get the Doo-Sabin subdivision surface, either we look for other shapes (either smooth and differentiable or not). Figure 12

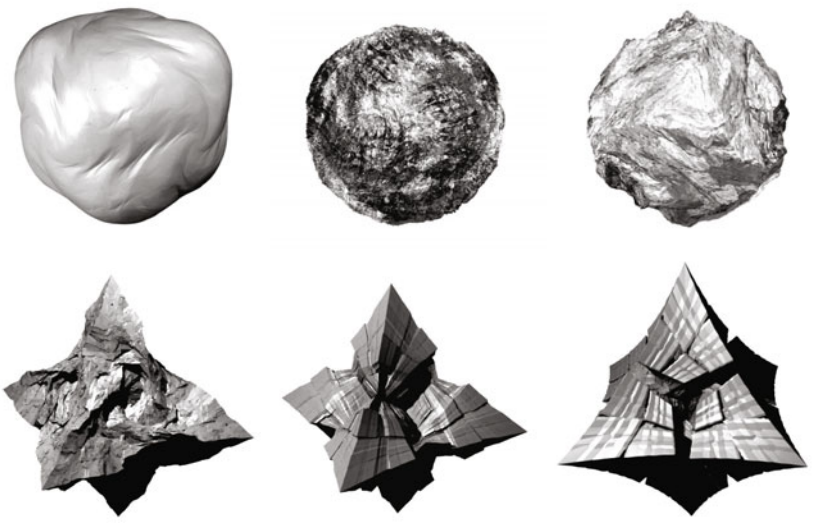


Fig. 12. Examples of different "*geometric textures*" obtained obtained by subdividing a cube with different subdivision weights

shows six different surfaces obtained by subdividing a cube with different triples of weights.

In this paper we have presented how to model curves and surfaces by means of iterative process, namely linear BC-IFS (Boundary Controled Iteratif Function System). This approach guarantees the required topology of the final shape by introducing incidence and adjacency constraints on a B-Rep model. For an linear BC-IFS it implies constraints on underlying matrices representing subdivision operators.

In this paper we have explicitly constructed local corner cutting curves and studied the differential behaviour by analyzing eigenvalues and eigenvectors of the subdivision operators. While we find same necessary conditions as do Gregory and De Boor, we study a larger family of curves, since by using BC-IFS approach we are able to enrich the convergence domain, thus introducing new shapes. To characterize different families of shapes in the convergence domain we study eigenstructures of subdivision operators and propose a precise cartography of all the regions.

We have also proved that necessary conditions are also sufficient ones. Moreover, we have proved that stationary local corner cutting curves are differentiable *almost everywhere.*

Finally, we have proposed a new roughness descriptor of fractal shapes. With this descriptor it is immediate to see where in the convergence domain we have to look for rough or smooth curves. Indeed, even if a curve is not differentiable it may look very smooth.

Acknowledgment. This work has been supported by French National Research Agency (ANR) through COSINUS program (project MODITERE n° ANR-09-COSI-014).

References

1. Barnsley, M.: Fractals everywhere. Academic Press Professional, Inc., San Diego (1988)
2. Bensoudane, H.: Étude différentielle des formes fractales. PhD thesis, Université de Bourgogne (2009)
3. Bensoudane, H., Gentil, C., Neveu, M.: Fractional half-tangent of a curve described by iterated function system. Journal Of Applied Functional Analysis 4(2), 311–326 (2009)
4. Boier-Martin, I., Zorin, D.: Differentiable parameterization of Catmull-Clark subdivision surfaces. In: SGP 2004: Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, pp. 155–164. ACM, New York (2004)
5. De Boor, C.: Local corner cutting and the smoothness of the limiting curve. Computer Aided Geometric Design 7(5), 389–397 (1990)
6. Cochran, W.O., Lewis, R.R., Hart, J.C.: The normal of a fractal surface. The Visual Computer 17(4), 209–218 (2001)
7. Gentil, C.: Les fractales en synthèse d'image: le modèle IFS. PhD thesis, Université LYON 1 (1992)
8. Gregory, J.A., Qu, R.: Nonuniform corner cutting. Computer Aided Geometric Design 13(8), 763–772 (1996)
9. Hutchinson, J.: Fractals and self-similarity. Indiana University Journal of Mathematics 30(5), 713–747 (1981)
10. Khodakovsky, A., Schröder, P.: Fine level feature editing for subdivision surfaces. In: SMA 1999: Proceedings of the Fifth ACM Symposium on Solid Modeling and Applications, pp. 203–211. ACM, New York (1999)
11. Kolwankar, K.M., Gangal, A.D.: Hölder exponents of irregular signals and local fractional derivatives. Pramana 48(1), 49–68 (1997)
12. Kolwankar, K.M., Gangal, A.D.: Local fractional derivatives and fractal functions of several variables. In: Proc. of Fractals in Engineering (1997)
13. Daniel Mauldin, R., Williams, S.C.: Hausdorff dimension in graph directed constructions. Transactions of the American Mathematical Society 309(2), 811–829 (1988)
14. Paluszny, M., Prautzsch, H., Schäfer, M.: A geometric look at corner cutting. Computer Aided Geometric Design 14(5), 421–447 (1997)
15. Prusinkiewicz, P., Hammel, M.S.: Language-restricted iterated function systems, koch constructions and l-systems. In: New Directions for Fractal Modeling in Computer Graphics, ACM SIGGRAPH Course Notes. ACM Press, New York (1994)
16. Scealy, R.: V -variable fractals and interpolation. PhD thesis, Australian National University (2008)
17. Sokolov, D., Gentil, C.: Sufficient conditions for differentiability of local corner cutting curves. Research report, LE2I, Université de Bourgogne (2010)

18. Stam, J.: Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values. In: Proceedings of SIGGRAPH, pp. 395–404 (1998)
19. Tosan, E., Bailly-Sallins, I., Gouaty, G., Stotz, I., Buser, P., Weinand, Y.: Une modélisation géométrique itérative basée sur les automates. In: GTMG 2006, Journées du Groupe de Travail en Modélisation Géométrique, Cachan, March 22-23, pp. 155–169 (2006)
20. Zair, C.E., Tosan, E.: Fractal modeling using free form techniques. Computer Graphics Forum 15(3), 269–278 (1996)

Algebraic Curves of Low Convolution Degree

Jan Vršek and Miroslav Lávička

University of West Bohemia, Faculty of Applied Sciences,
Department of Mathematics, Plzeň, Czech Republic
{vrsekjan,lavicka}@kma.zcu.cz

Abstract. Studying convolutions of hypersurfaces (especially of curves and surfaces) has become an active research area in recent years. The main characterization from the point of view of convolutions is their convolution degree, which is an affine invariant associated to a hypersurface describing the complexity of the shape with respect to the operation of convolution. Extending the results from [1], we will focus on the two simplest classes of planar algebraic curves with respect to the operation of convolution, namely on the curves with the convolution degree one (so called LN curves) and two. We will present an algebraic analysis of these curves, provide their decomposition, and study their properties.

1 Introduction

In Computer Aided Geometric Design, the convolution hypersurface $\mathcal{V} \star \mathcal{W}$ of two hypersurfaces $\mathcal{V}$, $\mathcal{W}$ is introduced as the envelope of copies of $\mathcal{V}$ obtained by the translations along vectors from $\mathcal{W}$. A fundamental characteristic is the so called convolution degree. This is an affine invariant associated to any hypersurface which determines the complexity of the given hypersurface with respect to the operation of convolution. The higher the convolution degree is, the more complicated resulting objects can be obtained. Hence, current methods and algorithms use mainly hypersurfaces of low convolution degree, basically one or two. The former case are well-known LN hypersurfaces (hypersurfaces with Linear Normals) and the examples of latter hypersurfaces are circles, ellipses, hyperbolas, etc. (curve case) and spheres, ellipsoids, hyperboloids, etc. (surface case). Therefore studying convolutions of hypersurfaces of convolution degree one and two involves, among others, also studying offsets as special and most prominent cases. Hence, the theory of convolutions is closely related to the theory of PH/MPH curves and PN/MOS (hyper)surfaces, too. The reader interested in these topics is kindly referred to [2,3,4,5,6,7,8,9] and references therein.

As concerns convolutions, hypersurfaces with Linear field of Normal vectors (LN hypersurfaces), introduced in [10], possess the biggest application potential, as they admit rational convolutions with any arbitrary rational hypersurface (cf. [11,12]). However, the convolution of two rational hypersurfaces is not rational in general, cf. [13,14] and references therein for more details. This inconvenience occurs already for hypersurfaces of convolution degree two where the so-called RC properties (guaranteeing Rational Convolutions) must be taken into account.

A novel and very beneficial approach for dealing with convolutions using the so-called support functions was introduced in [15,16]. In particular, it was shown in [16] that odd rational support functions correspond to those rational hypersurfaces which can be equipped with a linear field of normal vectors.

In this paper, we restrict ourselves to the curve case and develop further the study which started in [1], where a thorough algebraic analysis of convolutions was provided and a formula relating the convolution degree and the genus of an algebraic curve was derived. The ideas presented in [1] extended approaches devoted to offsets from [17,18,19]. Recently, it has been showed that many examples of curves of convolution degree one and two can be found among hypocycloids and epicycloids (HE-cycloids), cf. [20]. This fact was later used for formulation of G^1 Hermite interpolation algorithm based on HE-splines. We can also find other interpolation/approximation methods based on LN curves or curves with convolution degree two (especially PH curves) – see e.g. [21,22,14,23]. Thus, one can ask what new may be found about these well-known and thoroughly studied geometric objects.

The main contribution of this paper is a thorough algebraic analysis of planar algebraic curves of convolution degree one and two. We present their algebraic and geometric properties and provide their decomposition based on the set of unique fundamental generators. A main part will be devoted to rational curves equipped with a quadratic field of normal vectors, for which the possible associated RC conditions will be analyzed. This characterization uses the so-called generalized Blaschke cylinder as a generalization of the well-known concept used for PH curves. Clearly, this analysis is necessary for formulating potential subsequent algorithms.

2 Preliminaries

The theory presented in this paper is based on fundamentals of algebraic geometry of planar curves working over the field of complex numbers $\mathbb{C}$; see e.g. [24,25,26] for more details. On the other hand, problems originating in geometric modeling work especially over the field of real numbers and thus some difficulties may appear and certain algebraic geometry techniques must be reconsidered when used in particular applications. However, using the field of complex numbers is necessary to obtain global results dealing with convolutions of algebraic curves.

2.1 Convolution of Algebraic Curves

There is only one tangent direction for all points on the line and this may cause some troubles when formulating general theorems concerning convolutions. Nevertheless, constructing convolutions of straight lines with any curve is trivial (as it can be seen from Definition 2), so we can omit this case. Hence, from now on if we say a curve we implicitly assume that this is not a line.

Definition 1. *Let $\mathcal{V}, \mathcal{W} \subset \mathbb{C}^2$ be algebraic curves. Then two points $\mathbf{v} \in \mathcal{V}$ and $\mathbf{w} \in \mathcal{W}$ are said to be* coherent*, denoted by $(\mathbf{v}, \mathcal{V}) \sim_\star (\mathbf{w}, \mathcal{W})$ if the following two conditions are fulfilled:*

1. *$\mathbf{v}$ and $\mathbf{w}$ are regular points,*
2. *$\mathrm{T}_\mathbf{v}\mathcal{V} \parallel \mathrm{T}_\mathbf{w}\mathcal{W}$, where $\mathrm{T}_\mathbf{x}\mathcal{X}$ denotes the tangent line to the curve $\mathcal{X}$ at the point $\mathbf{x} \in \mathcal{X}$.*

Using the notion of coherent points, we can define the convolution of two algebraic curves, denoted by $\star$.

Definition 2. *The* convolution $\mathcal{V} \star \mathcal{W}$ *of curves $\mathcal{V}, \mathcal{W} \subset \mathbb{C}^2$ is defined as the smallest algebraic set containing*

$$\left\{\mathbf{v} + \mathbf{w} \in \mathbb{C}^2 | (\mathbf{v}, \mathcal{V}) \sim_\star (\mathbf{w}, \mathcal{W})\right\}. \tag{1}$$

According to the previous definition, the convolution of two curves is an algebraic set. Nonetheless, it may degenerate to a point (i.e., to a zero-dimensional algebraic set) in some cases, and moreover it may be reducible despite the fact that both input curves are irreducible. The following definition describes different types of possible components of $\mathcal{V} \star \mathcal{W}$, see also [17].

Definition 3. *An irreducible component $\mathcal{U} \subset \mathcal{V} \star \mathcal{W}$ is called* simple*,* k-special*, or* degenerated*, if there exists a dense set with respect to Zariski topology $U \subset \mathcal{U}$ such that for all $\mathbf{u} \in U$ there exist(s) unique, exactly $1 < k < \infty$, or infinitely many pair(s) $\mathbf{v} \in \mathcal{V}$, $\mathbf{w} \in \mathcal{W}$ such that $(\mathbf{v}, \mathcal{V}) \sim_\star (\mathbf{w}, \mathcal{W})$, respectively, and $\mathbf{u} = \mathbf{v} + \mathbf{w}$.*

The construction of the convolution curve of two given curves is shown in Fig. 1 (left), where the dashed component is 2-special and the dotted one is simple. If there is no danger of confusion, we will use only the notion a *special component* instead of k-special. Since $(\mathcal{U} \cup \mathcal{V}) \star \mathcal{W} = (\mathcal{U} \star \mathcal{W}) \cup (\mathcal{V} \star \mathcal{W})$ by the definition of convolution, we will assume throughout the paper that the input curves are irreducible.

The most important number associated to any curve from the point of view of convolutions is the so-called convolution degree, which measures the complexity of a given curve with respect to the operation of convolution.

Definition 4. *The* convolution degree $\kappa_\mathcal{V}$ *of a planar curve $\mathcal{V}$ is equal to the number of affine regular points on $\mathcal{V}$ where the tangent lines are parallel with a fixed generic direction.*

It immediately follows from the definition that the convolution degree is an affine invariant of the curve.

2.2 Previous Work on Convolutions

Since the curves in CAGD are usually given by its rational representations it is natural to look for a parameterization of some component of $\mathcal{V} \star \mathcal{W}$. This approach leads to the so-called *parameter varieties* introduced in [27].

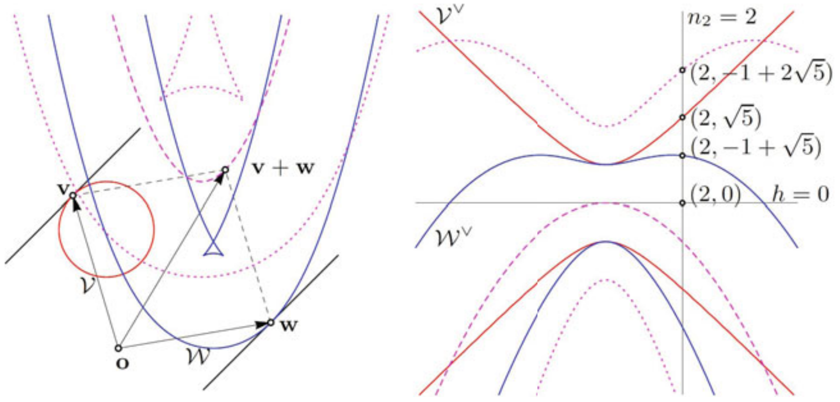


Fig. 1. The construction of the convolution curve in the primal (left) and dual (right) plane (the dual image is dehomogenized by setting $n_1 = 1$) – the red and blue curves are the irreducible input curves, their reducible convolution is shown in purple (dotted simple component and dashed special component)

Definition 5. *Two parameterizations* $\mathbf{v}(t)$ *and* $\mathbf{w}(t)$ *are said to be* coherent*, written* $\mathbf{v}(t) \sim_\star \mathbf{w}(t)$*, if for a generic* $t_0 \in \mathbb{C}$ *it holds* $(\mathbf{v}(t_0), \mathcal{V}) \sim_\star (\mathbf{w}(t_0), \mathcal{W})$*.*

If $\mathbf{v}(t) \sim_\star \mathbf{w}(t)$ then $\mathbf{v}(t) + \mathbf{w}(t)$ parameterizes some component of $\mathcal{V} \star \mathcal{W}$. Of course, two generic parameterizations are not coherent and thus the main problem of the parametric approach can be formulated as follows: For given parameterizations $\mathbf{v}(s)$ and $\mathbf{w}(t)$ find rational functions $\varphi, \psi \in \mathbb{C}(u)$ such that $\mathbf{v}(\varphi(u)) \sim_\star \mathbf{w}(\psi(u))$.

Then the parameter variety can be defined as a curve in the space of parameters given by the relation

$$\{(s,t) \in \mathbb{C} \times \mathbb{C} \mid (\mathbf{v}(s), \mathcal{V}) \sim_\star (\mathbf{w}(t), \mathcal{W})\}. \tag{2}$$

The significance of the parameter variety is established by the following proposition which follows immediately from the above definitions.

Proposition 1. *Let* $(\varphi(u), \psi(u))$ *be a parameterization of a component of the parameter variety* (2) *associated to* $\mathbf{v}(s)$ *and* $\mathbf{w}(t)$*. Then* $\mathbf{v}(\varphi(u)) + \mathbf{w}(\psi(u))$ *parameterizes a component of* $\mathcal{V} \star \mathcal{W}$*.*

Moreover, one can easily obtain a defining equation of the parameter variety. Let us denote

$$\left(\frac{\hat{p}_1}{\hat{p}_0}, \frac{\hat{p}_2}{\hat{p}_0}\right) = \frac{\mathrm{d}\mathbf{v}}{\mathrm{d}s} \qquad \text{and} \qquad \left(\frac{\hat{q}_1}{\hat{q}_0}, \frac{\hat{q}_2}{\hat{q}_0}\right) = \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t}. \tag{3}$$

Then

$$p_i := \frac{\hat{p}_i}{\gcd(\hat{p}_1, \hat{p}_2)} \qquad \text{and} \qquad q_i := \frac{\hat{q}_i}{\gcd(\hat{q}_1, \hat{q}_2)}, \qquad i = 1, 2, \tag{4}$$

are polynomials in s and t, respectively, such that the defining polynomial of the parameter variety is given by

$$c_{\mathbf{v},\mathbf{w}}(s,t) := p_1(s)q_2(t) - p_2(s)q_1(t). \tag{5}$$

Let us denote that the algebraic degree of $c_{\mathbf{v},\mathbf{w}}$ is closely related to the convolution degree of the curves. In particular, if $\mathbf{v}(s)$ is a proper parameterization (i.e., a birational mapping $\mathbb{C} \to \mathcal{V}$) then we have

$$\kappa_{\mathcal{V}} = \deg_s c_{\mathbf{v},\mathbf{w}}(s,t) = \max\{\deg p_1(s), \deg p_2(s)\}. \tag{6}$$

In addition, convolutions have considerably simple descriptions if we apply the dual approach. Recall that a curve $\mathcal{X} : f(\mathbf{x}) = 0$ has the *dual representation*

$$\mathcal{X}^\vee : \ F^\vee(\mathbf{n}, h) = 0, \tag{7}$$

where $F^\vee$ is a homogeneous polynomial in $\mathbf{n} = (n_1, n_2)$ and h. The set of all lines

$$\left\{\mathbf{x} \in \mathbb{C}^2 \,|\, \mathbf{n} \cdot \mathbf{x} = h \quad \text{and} \quad F^\vee(\mathbf{n}, h) = 0\right\} \tag{8}$$

forms a system of *tangent lines* of $\mathcal{X}$ with the normal vectors $\mathbf{n}$, cf. [28].

Let $\mathbf{v} \in \mathcal{V}$ and $\mathbf{w} \in \mathcal{W}$ be two generic coherent points, and $T_{\mathbf{v}}\mathcal{V}$ and $T_{\mathbf{w}}\mathcal{W}$ be the tangent lines at these points given by

$$\mathbf{n} \cdot \mathbf{x} = h_1, \quad \text{and} \quad \mathbf{n} \cdot \mathbf{x} = h_2, \tag{9}$$

respectively. Then the tangent line at the point $\mathbf{v} + \mathbf{w} \in \mathcal{V} \star \mathcal{W}$ has the equation

$$\mathbf{n} \cdot \mathbf{x} = h_1 + h_2 = h_3. \tag{10}$$

Hence, we arrive at the following. Let the curves $\mathcal{V}$ and $\mathcal{W}$ have dual defining polynomials $F^\vee(\mathbf{n}, h)$ and $G^\vee(\mathbf{n}, h)$, respectively. Then we compute the dual equation of $\mathcal{V} \star \mathcal{W}$ by eliminating (e.g. using Gröbner bases) variables h_1 and h_2 from the system of equations

$$F^\vee(\mathbf{n}, h_1) = 0, \qquad G^\vee(\mathbf{n}, h_2) = 0 \qquad \text{and} \qquad h_3 - h_1 - h_2 = 0. \tag{11}$$

A visualization of the dual approach is demonstrated in Fig. 1 (right), where one can see the special character of the dashed component better than in the primal case.

Moreover the convolution degree is naturally involved in the dual approach as it holds

$$\kappa_{\mathcal{V}} = \deg_h F^\vee(\mathbf{n}, h) \tag{12}$$

for the curve $\mathcal{V}$ given dually by $F^\vee(\mathbf{n}, h) = 0$, cf. [29,20].

3 Curves of Convolution Degree One

Clearly, the simplest curves with respect to the operation of convolution are the curves with convolution degree one. Fundamental facts concerning these curves were thoroughly studied in [1], see this reference for more details (especially Theorem 5.4 and Theorem 5.5). In this paper we extend our observations and provide a decomposition of an arbitrary LN curve into the convolution of a finite number of suitable fundamental curves.

Definition 6. *Any curve $\mathcal{V}$ with $\kappa_{\mathcal{V}} = 1$ will be called an* LN curve, *where LN stands for* linear normals. *The set of all LN curves will be denoted by $\mathfrak{L}$.*

Although this approach is natural when working with convolutions, the original definition is different. However, one can prove that all LN curves in our meaning are rational and fulfill the distinguished condition from [10], and vice versa.

For the sake of convenience, we extend the definition by allowing also convolutions with points

$$\mathbf{p} \star \mathcal{V} = \{\mathbf{x} \in \mathbb{C}^2 \mid \exists \mathbf{v} \in \mathcal{V} : \mathbf{x} = \mathbf{v} + \mathbf{p}\}, \qquad \mathbf{p} \star \mathbf{q} = \mathbf{p} + \mathbf{q}, \tag{13}$$

where $\mathbf{p}, \mathbf{q} \in \mathbb{C}^2$. Then one can prove that the set of all algebraic sets in $\mathbb{C}^2$ of dimension 0 or 1 along with the operation of convolution forms a commutative monoid with the neutral element $\mathbf{o} = (0, 0)$. In addition, the set of all LN curves together with one-point sets is its maximal subgroup.

A better insight into the properties of this group can be gained if we describe the set of its generators. In what follows, we will show that this is equivalent to a construction of the full decomposition of LN curves into the convolutions of some fundamental LN curves.

Theorem 1. *Any LN curve can be obtained as the convolution of a finite number of affine images of the canonical curves $x_1^k - x_2^{k+1} = 0$.*

Proof. By (12), the dual equation of an LN curve has the form

$$F^{\vee}(\mathbf{n}, h) = f_{m-1}(\mathbf{n})h + f_m(\mathbf{n}), \tag{14}$$

where f_i is a homogeneous polynomial of degree i, and hence

$$h = -\frac{f_m(\mathbf{n})}{f_{m-1}(\mathbf{n})}. \tag{15}$$

For the sake of simplicity, we assume that n_1 divides neither $f_{m-1}(\mathbf{n})$, nor $f_m(\mathbf{n})$. Thus after dehomogenization $n = n_2/n_1$ we may write the decomposition of (15) into the partial fractions

$$h = h_0 + \cdots + h_\ell, \quad \text{where} \quad h_i = \frac{\alpha_i n_1^{k+1}}{(\beta_i n_1 + \gamma_i n_2)^k} \tag{16}$$

for some $\alpha_i, \beta_i, \gamma_i \in \mathbb{C}$ and $k \in \mathbb{N}$. Then the transformation $n_1' = \alpha_i n_1$, $n_2' = \beta_i n_1 + \gamma_i n_2$ and $h' = h$ induces an affine transformation which maps a curve described by h_i to the desired canonical form. $\square$

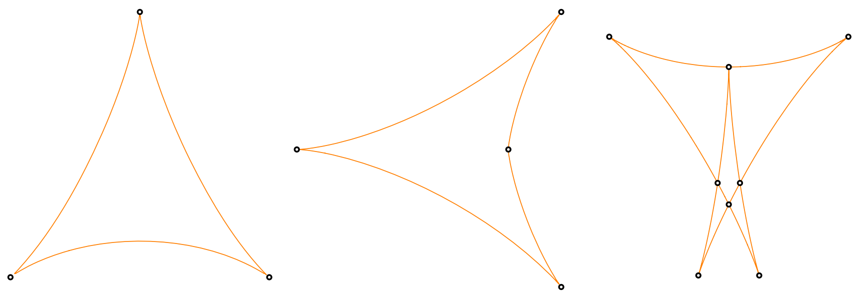


Fig. 2. Fundamental LN curves from Remark 1; $k = 2$, $i = 0, 1, 2$ (the black dots denote real singularities)

Remark 1. Let us emphasize that despite starting with a real LN curve, it may happen that some of the factors (16) possess non-real coefficients and hence the induced transformation does, too. So, if we prefer to work over the reals we have to introduce new real fundamental curves corresponding to irreducible factors. In particular, these LN curves are given by the dual equations

$$(n_1^2 + n_2^2)^k h - n_1^i n_2^{2k+1-i} = 0, \tag{17}$$

where $k \geq 1$ and $0 \leq i \leq k$. For instance, for $k = 1$ and $i = 0$ we obtain the well-known hypocycloid called deltoid (or tricuspoid). The fundamental LN curves for $k = 2$ are shown in Fig. 2.

4 Curves of Convolution Degree Two

Throughout this section, we consider by $\mathcal{V}$ a curve with the convolution degree two. We focus mainly on the subset of rational curves, study thoroughly their properties and, analogously to the LN curves, provide their decomposition.

4.1 Elementary Properties of Curves with Convolution Degree Two

By (12), the dual equation of curves of the convolution degree 2 is of the form

$$F^{\vee}(\mathbf{n}, h) = f_{m-2}(\mathbf{n})h^2 + f_{m-1}(\mathbf{n})h + f_m(\mathbf{n}). \tag{18}$$

Thus any curve of this type is square-root parameterizable, i.e., it can be parameterized in terms of t and $\sqrt{P(t)}$, where $P(t)$ is a polynomial in t. This shows that the curves of convolution degree 2 need not to be rational in general (in contrast to LN curves studied in the previous section). In particular, it is known that the only curves which admit square-root parameterizations are rational, elliptic or hyper-elliptic, cf. [30,31] for further details.

Proposition 2. *Any curve with convolution degree 2 is rational, elliptic, or hyper-elliptic.*

Due to their greater convolution degree, the resulting convolutions are generally more complicated than convolutions with LN curves. Nevertheless, we can still find analogous results, see [1] (especially Theorem 5.6 and Theorem 5.7).

Remark 2. Since for two coherent points $(\mathbf{v}, \mathcal{V}) \sim_\star (\mathbf{w}, \mathcal{W})$ the point $\mathbf{v} + \mathbf{w} \in \mathcal{V} \star \mathcal{W}$ is singular or coherent to both points $\mathbf{v}$ and $\mathbf{w}$, see [1], we can easily compute the convolution degree of the components of $\mathcal{V} \star \mathcal{W}$. In particular, if $\mathcal{V} \star \mathcal{W}$ is irreducible we have $\kappa_{\mathcal{V}\star\mathcal{W}} = 2\kappa_{\mathcal{W}}$ and in the case of reducible convolution we arrive at $\kappa_{\mathcal{X}} = \kappa_{\mathcal{W}}$ for the simple component $\mathcal{X}$ and $\kappa_{\mathcal{U}} = \frac{1}{2}\kappa_{\mathcal{W}}$ for the 2-special component $\mathcal{U}$.

A main disadvantage of curves of the convolution degree 2, despite being rational, is that the convolution $\mathcal{V} \star \mathcal{W}$ is not rational in general. However, the situation becomes considerably simpler when $\mathcal{V} \star \mathcal{W}$ is reducible, cf. [1, Theorem 5.7].

Theorem 2. *If $\mathcal{V} \star \mathcal{W}$ possesses two components then each simple component is birationally equivalent to $\mathcal{W}$. Moreover if $\mathcal{U} \subset \mathcal{V} \star \mathcal{W}$ is special then* $\mathrm{g}(\mathcal{U}) \leq \mathrm{g}(\mathcal{W})$, *where* $\mathrm{g}(\mathcal{X})$ *stands for the genus of a curve $\mathcal{X}$.*

In particular, if the curve $\mathcal{W}$ in Theorem 2 is rational then each non-degenerated component is rational, too. Moreover by Theorem 5.6 in [1] the curve $\mathcal{V}$ has to be also rational and there exists a proper parameterization $\mathbf{v}(s)$. Then for an arbitrary parameterization $\mathbf{w}(t)$ there exists a rational function $\varphi(t)$ of degree $\frac{1}{2}\kappa_{\mathcal{W}} \cdot \deg \mathbf{w}$ such that $\mathbf{v}(\varphi(t)) \sim_\star \mathbf{w}(t)$.

Moreover, if a proper parameterization $\mathbf{w}(t)$ is $\mathcal{V}$-coherent then $\mathcal{V} \star \mathcal{W}$ is reducible. To show this, consider $\mathcal{V} \star \mathcal{W}$ to be irreducible and choose a generic point $\mathbf{w}$ on $\mathcal{W}$. Then there exist exactly two coherent points $\mathbf{v}_1$, $\mathbf{v}_2$ on $\mathcal{V}$. Since $\mathcal{V} \star \mathcal{W}$ is irreducible it is simple (again by Theorem 5.6 in [1]). Hence, to obtain points $\mathbf{v}_1 + \mathbf{w}$ and $\mathbf{v}_2 + \mathbf{w}$ we have to trace the curve $\mathcal{W}$ twice.

Finally, if $\mathcal{V}$ is rational then looking at the formula for the convolution degree of parametric curves, cf. (6), we see that its normal vectors depend on the parameter quadratically. This motivates us to the following definition which extends the similar idea used for the LN curves.

Definition 7. *Any rational curve with the convolution degree 2 will be called a* QN curve, *where QN stands for* quadratic normals. *The set of all QN curves will be denoted by $\mathfrak{Q}$.*

Clearly, QN curves play among all algebraic curves with the convolution degree 2 the most important role, mainly due to their potential applications in technical praxis as NURBS curves.

4.2 Rationality of Convolutions with QN Curves

In what follows, we consider $\mathcal{V}$ to be a QN curve. Let $\mathbf{v}(s) : \mathbb{C} \to \mathcal{V}$ be its proper parameterization and $p_1(s), p_2(s) \in \mathbb{C}[s]$ be the associated polynomials as in (4). As $\kappa_{\mathcal{V}} = 2$ the maximum of the degrees of $p_i(s)$ is 2, see (6). Hence

$$p_1(s)x + p_2(s)y = (s^2,\, s,\, 1) \cdot (\hat{\mathbf{p}}_1 x + \hat{\mathbf{p}}_2 y) = 0 \tag{19}$$

is a quadratic equation in s with the coefficients from $\mathbb{C}[x, y]$. The discriminant of this equation plays a key role in deciding the rationality of convolutions with the QN curve $\mathcal{V}$. Its fundamental properties are summarized in the next lemma, proof of which can be easily obtained by a direct computation.

Lemma 1. *Let the discriminant of* (19) *be denoted by* $D_{\mathbf{v}}(x, y) = D_{\mathbf{v}}(\mathbf{x})$*. Then it holds*

(i) $D_{\mathbf{v}}(\mathbf{x}) = \mathbf{x} \cdot \Delta \cdot \mathbf{x}^T$*, where the elements of the matrix* Δ *are computed as* $\Delta_{ij} = \hat{\mathbf{p}}_i^T \cdot \begin{pmatrix} 0 & 0 & -2 \\ 0 & 1 & 0 \\ -2 & 0 & 0 \end{pmatrix} \cdot \hat{\mathbf{p}}_j$ *and* $\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2$ *are from* (19)*,*

(ii) $\det \Delta \neq 0$*,*

(iii) Let $A \in \mathbf{GL}_2(\mathbb{C})$ *and* $\mathbf{b} \in \mathbb{C}^2$*. Then* $D_{A\mathbf{v}+\mathbf{b}} = \mathbf{x} \cdot \left(A \cdot \Delta \cdot A^T\right) \cdot \mathbf{x}^T$*.*

(iv) For any $\varphi(t) = \frac{\alpha_1 t + \alpha_2}{\beta_1 t + \beta_2}$ *we have* $D_{\mathbf{v}\circ\varphi}(\mathbf{x}) = (\alpha_1\beta_2 - \alpha_2\beta_1)^2 \cdot D_{\mathbf{v}}(\mathbf{x})$*.*

Remark 3. It follows from Lemma 1 (iv) that for two different proper parameterizations of the same curve the corresponding discriminants differ only by a multiple of some non-zero constant. Hence, the discriminant may be always normalized using a suitable φ such that the leading coefficient with respect to the lexicographic order equals 1. In what follows, this normalized discriminant of the quadratic equation (19) will be denoted by D_Δ.

It turns out that QN curves with the same D_Δ behave similarly with respect to the operation of convolution. Moreover, there is a natural representative of the set of all QN curves with the same D_Δ having the dual equation $D_\Delta(\mathbf{n}) - h^2 = 0$.

Definition 8. *A regular conic section given by the dual representation*

$$D_\Delta(\mathbf{n}) - h^2 = 0 \tag{20}$$

is called a canonical conic section *of the class of QN curves characterized by* D_Δ*. We will denoted it by* $\mathcal{S}^\Delta$*. Next, the set of all QN curves with the same* D_Δ *will be denoted by* $\mathfrak{Q}_\Delta$*.*

Example 1. Let $\mathcal{T}$ be the Tschirnhausen cubic given parameterically by

$$\mathbf{t}(s) = \left(3(s^2 - 3), s(s^2 - 3)\right)^T. \tag{21}$$

Then $p_1(s) = 6s$ and $p_2(s) = 3(s^2 - 1)$ and the corresponding discriminant is equal to $D_{\mathbf{t}}(\mathbf{x}) = 36(x^2 + y^2)$. Next, we arrive at $D_\Delta(\mathbf{x}) = x^2 + y^2$ and the canonical conic section has the dual equation $D_\Delta(\mathbf{n}) - h^2 = n_1^2 + n_2^2 - h^2 = 0$. Since the primal curve is defined by the equation $x^2 + y^2 - 1 = 0$, the class of QN curves containing the Tschirnhausen cubic is represented by the unit circle.

Taking into account that for any symmetric matrix $B \in \mathbf{GL}_2(\mathbb{C})$ there exists the matrix $A \in \mathbf{GL}_2(\mathbb{C})$ such that $A \cdot B \cdot A^T$ is the identity matrix and invoking Lemma 1 (iii) we obtain the lemma:

Lemma 2. $\mathbf{GL}_2(\mathbb{C})$ *acts transitively on the set of all classes of QN curves.*

Remark 4. Obviously, the classes $\mathfrak{Q}_\Delta$ are not preserved under affine transformations. Nonetheless, for any arbitrary curve $\mathcal{V} \in \mathfrak{Q}_\Delta$ there exists a linear transformation A such that $A(\mathcal{V})$ is a PH curve. However, let us emphasize that when working only with real curves, it is not generally ensured that for a real curve $\mathcal{V}$ the curve $A(\mathcal{V})$ is real, too. In particular, any real QN curve can be transformed either to a real PH curve, or to a real MPH curve (the class of MPH curves is determined by the canonical conic section $x^2 - y^2 - 1 = 0$, cf. [5,6,32]).

In what follows, we will show that all curves from the same class $\mathfrak{Q}_\Delta$ behave similarly with respect to the operation of convolution.

Lemma 3. *For a curve $\mathcal{V} \in \mathfrak{Q}_\Delta$ and an arbitrary rational curve $\mathcal{W}$, it holds that the parameterization $\mathbf{w}(t)$ is $\mathcal{V}$-coherent if and only if $D_\Delta(\mathbf{n}(t)) = \sigma^2(t)$ for some $\sigma \in \mathbb{C}(t)$, where $\mathbf{n}(t) = (\mathrm{d}\,\mathbf{w}(t)/\mathrm{d}\,t)^\perp$ is the normal vector field of $\mathcal{W}$.*

Proof. We choose an arbitrary proper parameterization $\mathbf{v}(s)$. Then comparing (5) and (19), it is seen that $\mathbf{w}(t)$ is $\mathcal{V}$-coherent if and only if there exists $\varphi(t) \in \mathbb{C}(t)$ such that

$$p_1(\varphi(t))n_1(t) + p_2(\varphi(t))n_2(t) \equiv 0. \tag{22}$$

Hence, for a given $\mathcal{V}$-coherent parameterization $\mathbf{w}(t)$ we can find a rational function $\varphi(t)$ such that for all t_0 the corresponding value $\varphi(t_0)$ is a root of the polynomial

$$p_1(s)n_1(t_0) + p_2(s)n_2(t_0) = c_2(\mathbf{n}(t_0))s^2 + c_1(\mathbf{n}(t_0))s + c_0(\mathbf{n}(t_0)). \tag{23}$$

Thus we obtain

$$\varphi(t) = \frac{-c_1(\mathbf{n}(t)) \pm \sqrt{c_1^2(\mathbf{n}(t)) - c_2(\mathbf{n}(t))c_0(\mathbf{n}(t))}}{2c_2(\mathbf{n}(t))} = \frac{-c_1(\mathbf{n}(t)) \pm \sqrt{\alpha D_\mathcal{V}(\mathbf{n}(t))}}{2c_2(\mathbf{n}(t))}, \tag{24}$$

where α is a constant. Since $\varphi(t)$ is a rational function, there has to exist a rational function $\sigma(t)$ such that $D_\Delta(\mathbf{n}(t)) = \sigma^2(t)$.

Conversely, if $D_\Delta(\mathbf{n}(t))$ is a perfect square of some rational function then (24) defines a rational function $\varphi(t)$ reparameterizing $\mathbf{v}(s)$ such that $\mathbf{w}(t) \sim_\star \mathbf{v}(\varphi(t))$. $\square$

Remark 5. Let us recall, that conditions guaranteeing the rationality of the convolution $\mathcal{V} \star \mathcal{W}$ are called RC-conditions in [13]. Thus, $D_\Delta(\mathbf{n}(t)) = \sigma^2(t)$ is the RC-condition of a QN curve $\mathcal{V}$.

As an immediate consequence of the previous lemma, we obtain that for two curves $\mathcal{V}$ and $\mathcal{V}'$ in the same class $\mathfrak{Q}_\Delta$ the parameterization $\mathbf{w} \in \mathcal{W}$ is $\mathcal{V}$-coherent if and only if it is $\mathcal{V}'$-coherent. For instance (see Example 1) a parameterization $\mathbf{w}(t)$ is $\mathcal{T}$-coherent and $\mathcal{S}^1$-coherent if and only if it holds for the associated normal vector field $n_1^2(t) + n_2^2(t) = \sigma^2(t)$. This is nothing else than the well-known PH property.

As shown in [8], a parameterized curve fulfills the PH property if and only if its dual representation can be identified with a rational curve on the Blaschke cylinder $\mathcal{B}(n_1, n_2, h) : n_1^2 + n_2^2 = 1$. As we shall see bellow an analogous condition will be fulfilled for all classes of QN curves, too.

Definition 9. *Let us consider $D_\Delta(n_1, n_2)$ as a polynomial in $\mathbb{C}[n_1, n_2, h]$. Then the surface*

$$\mathcal{B}_\Delta : D_\Delta(n_1, n_2) - 1 = 0 \tag{25}$$

is called a generalized Blaschke cylinder*.*

Corollary 1. *If $\mathcal{V} \in \mathfrak{Q}_\Delta$ then there is a correspondence between $\mathcal{V}$-coherent parameterizations and rational curves on $\mathcal{B}_\Delta$.*

Proof. Let $\mathbf{w}(t)$ be a parameterization fulfilling the condition $D_\Delta(\mathbf{n}(t)) = \sigma^2(t)$. We set $h(t) = -\mathbf{w}(t) \cdot \mathbf{n}(t)$. Then, $\mathcal{W}$ can be considered as the envelope of one-parameter family of lines given by

$$\frac{\mathbf{n}(t)}{\sigma(t)} \cdot \mathbf{x} + \frac{h(t)}{\sigma(t)} = 0. \tag{26}$$

Hence, we arrive at the dual parameterization $\frac{1}{\sigma(t)}(\mathbf{n}(t), h(t))$ which is a rational curve on $\mathcal{B}_\Delta$.

Conversely let $(\mathbf{n}(t), h(t))$ be an arbitrary rational parameterized curve on $\mathcal{B}_\Delta$. Then, we can easily proceed to the representation of a curve as the envelope of one-parameter family of lines $\Sigma(t) : \mathbf{n}(t) \cdot \mathbf{x} + h(t) = 0$, where $D_\Delta(\mathbf{n}(t)) = \sigma^2(t)$. Solving the system $\Sigma(t) = 0, \Sigma_t(t) = 0$, where the subscript denotes the differentiation with respect to t, we arrive at the corresponding $\mathcal{V}$-coherent parameterization. $\square$

Remark 6. The above mentioned correspondence is not bijective but one-to-two. More precisely, both rational curves $\mathbf{x}(t) : \mathbb{C} \to \mathcal{B}_\Delta$ and $-\mathbf{x}(t) : \mathbb{C} \to \mathcal{B}_\Delta$ have the same image in the set of $\mathcal{V}$-coherent parameterizations, cf. Fig. 3. Hence, we have got the representation for all classes of QN curves analogous to the representation described in [8,9] for PH curves.

4.3 Decomposition of QN Curves

Similarly to the case of LN curves we would like to give a description of the set of QN curves. Since any QN curve lies in a certain class $\mathfrak{Q}_\Delta$ for some canonical conic section $\mathcal{S}^\Delta$, we will work with a fixed class $\mathfrak{Q}_\Delta$. Let us denote $\mathfrak{D}_\Delta = \mathbb{C}^2 \cup \mathfrak{L} \cup \mathfrak{Q}_\Delta$. Then, the following statement is obvious:

Lemma 4. *If $\mathcal{V}, \mathcal{W} \in \mathfrak{D}_\Delta$ then $\mathcal{V} \star \mathcal{W} \subset \mathfrak{D}_\Delta$.*

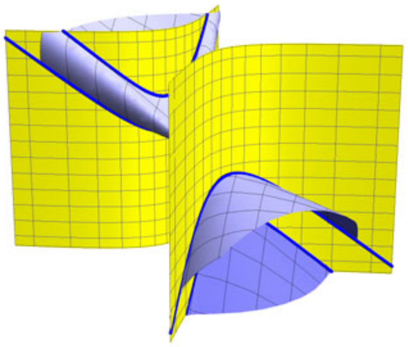
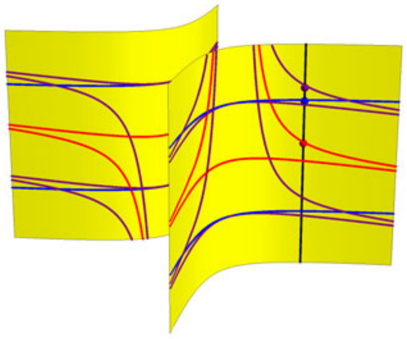



Fig. 3. Left: the generalized Blaschke cylinder $n_1^2 - n_2^2 - 1 = 0$ (yellow) with the image of a QN-curve (blue); Right: the construction of the convolution curve (purple) of two QN-curves (red and blue) solved on the associated generalized Blaschke cylinder

Hence, the set $\mathfrak{D}_\Delta$ is closed under the operation of convolution[1] and our goal is to find its generators, as in the case of LN curves. More precisely, we are going to find a family of fundamental curves $\{\mathcal{G}_\lambda\}_{\lambda\in\Lambda}$, where Λ is an (infinite) index set, such that an arbitrary curve $\mathcal{V} \in \mathfrak{Q}_\Delta$ is a component of

$$\left(\bigstar_{\lambda\in\Gamma} \mathcal{G}_\lambda\right) \star \mathcal{L} \star \mathbf{p}, \tag{27}$$

where $\mathcal{L} \in \mathfrak{L}$, $\mathbf{p} \in \mathbb{C}^2$ and $\Gamma \subset \Lambda$ is finite.

The dual equation of a general curve with the convolution degree 2 has the form

$$F^\vee(\mathbf{n}, h) = f_{m-2}(\mathbf{n})h^2 + f_{m-1}(\mathbf{n})h + f_m(\mathbf{n}) = 0, \tag{28}$$

where $f_i(\mathbf{n})$ are homogeneous polynomials of degree i. It is not difficult to realize that the set of all curves in $\mathfrak{Q}_\Delta$ such that $f_{m-1} = 0$ is closed under the operation of convolution. From the geometric point of view, the curves with this special dual representation are centrally symmetric.

Now, let $\mathcal{V}, \mathcal{W} \in \mathfrak{Q}_\Delta$ be two curves with the centers of symmetry $\mathbf{c}_1$ and $\mathbf{c}_2$, respectively. Then $\mathcal{V} \star \mathcal{W}$ is decomposed into two curves with the common center $\mathbf{c}_1 + \mathbf{c}_2$. As these curves play an important role in our further considerations, we will introduce the notation $\mathfrak{Q}^0_\Delta$ for them.

Lemma 5. *Any $\mathcal{V} \in \mathfrak{Q}$. can be written uniquely (up to a translation) as $\mathcal{V} = \mathcal{L} \star \mathcal{Q}$, where $\mathcal{L} \in \mathfrak{L}$ and $\mathcal{Q} \in \mathfrak{Q}^0_\Delta$.*

Proof. Let $\mathcal{V}$ has the dual equation $F^\vee(\mathbf{n}, h) = f_{m-2}(\mathbf{n})h^2 + f_{m-1}(\mathbf{n})h + f_m(\mathbf{n}) = 0$ and $\mathcal{L}'$ be an LN curve with the dual equation $g_{n-1}(\mathbf{n})h + g_n(\mathbf{n}) = 0$. For the

[1] For curves $\mathcal{V}, \mathcal{W} \in \mathfrak{Q}_\Delta \subset \mathfrak{D}_\Delta$ the convolution can be reducible and hence we have to write $\subset$ instead of $\in$ in Lemma 4.

sake of brevity, we will write f_i and g_i instead of $f_i(\mathbf{n})$ and $g_i(\mathbf{n})$. Using (11), one can compute the dual equation of $\mathcal{V} \star \mathcal{L}'$

$$f_{m-2}g_{n-1}^2h^2 + (2f_{m-2}g_n + f_{m-1}g_{n-1})g_{n-1}h + f_{m-2}g_n^2 + f_{m-1}g_{n-1}g_n + f_m g_{n-1}^2 = 0. \tag{29}$$

Hence, $\mathcal{V} \star \mathcal{L}' \in \mathfrak{Q}^0_\Delta$ if and only if $2f_{m-2}g_n + f_{m-1}g_{n-1} \equiv 0$. Let us write $f_{m-i} = f \cdot \tilde{f}_{m-i}$, for $i = 1, 2$, where $f = \gcd(f_{m-2}, f_{m-1})$ and set $g_{n-1} := -2\tilde{f}_{m-2}$ and $g_n := \tilde{f}_{m-1}$. Then $\mathcal{L}'$ is an LN curve such that $\mathcal{Q} = \mathcal{V} \star \mathcal{L}' \in \mathfrak{Q}^0_\Delta$ and denoting $\mathcal{L} = (\mathcal{L}')^-$ we arrive at $\mathcal{V} = \mathcal{Q} \star \mathcal{L}$.

Next, let $\mathcal{V} = \mathcal{Q} \star \mathcal{L} = \hat{\mathcal{Q}} \star \hat{\mathcal{L}}$. If $\hat{\mathcal{L}} \neq \mathcal{L}$ then $\mathcal{L} \star \hat{\mathcal{L}}^-$ is a LN curve such that

$$\mathcal{Q} = \hat{\mathcal{Q}} \star \left(\mathcal{L} \star \hat{\mathcal{L}}^-\right). \tag{30}$$

However, it is easy to see that the convolution of a curve in $\mathfrak{Q}^0_\Delta$ with an LN curve cannot lie in $\mathfrak{Q}^0_\Delta$, which is a contradiction. □

By Lemma 5 it is seen that solving the problem stated by (27) can be reduced to finding generators of the set $\mathfrak{Q}^0_\Delta$. The following lemma shows that the dual equations of curves from $\mathfrak{Q}^0_\Delta$ possess a special form.

Lemma 6. *$\mathcal{V}$ lies in $\mathfrak{Q}^0_\Delta$ if and only if the defining polynomial of $\mathcal{V}^\vee$ can be written in the form*

$$F^\vee(\mathbf{n}, h) = d_1(\mathbf{n})f^2(\mathbf{n})h^2 + d_2(\mathbf{n})g^2(\mathbf{n}), \tag{31}$$

where $d_1 \cdot d_2 = D_\Delta$.

Proof. $\mathcal{V} \in \mathfrak{Q}^0_\Delta$ if and only if it is a rational curve defined dually as $\mathcal{V}^\vee$: $f_{n-2}(\mathbf{n})h^2 + f_n = 0$ and its convolution with the corresponding canonical conic section $\mathcal{S}^\Delta$ is reducible.

First, let us assume that $\mathcal{V} \star \mathcal{S}^\Delta$ has a degenerated component. Then by Theorem 5.6 in [1] $\mathcal{V} = \mathcal{S}^{\Delta^-} = \mathcal{S}^\Delta$ and hence $F^\vee(\mathbf{n}, h) = h^2 + D_\Delta(\mathbf{n}, h)$.

Second, we will show that $\mathcal{V} \star \mathcal{S}^\Delta$ cannot have a special component. Recalling again Theorem 5.6 in [1], we get $\mathcal{V} = \mathcal{S}^\Delta \star \mathcal{L}$, where $\mathcal{L}$ has to be an LN curve since it holds $2 = \kappa_\mathcal{V} = \kappa_{\mathcal{S}^\Delta} \cdot \kappa_\mathcal{L} = 2\kappa_\mathcal{L}$. However as mentioned at the end of the proof of Lemma 5, the convolution of an LN curve and a curve from $\mathfrak{Q}^0_\Delta$ is not in $\mathfrak{Q}^0_\Delta$.

Thus, we may assume that both components of $\mathcal{V} \star \mathcal{S}^\Delta$ are simple and after computing the dual equation of the convolution we arrive at

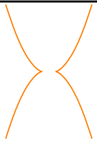
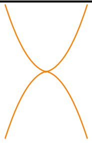
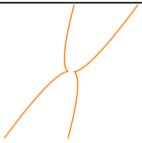
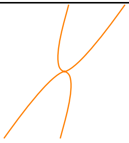
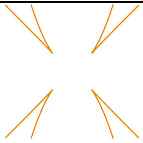
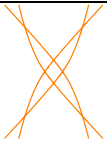
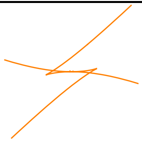
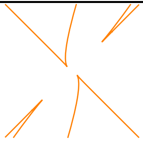
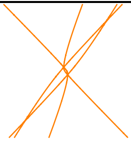
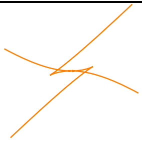
$$\begin{aligned}(f_{n-2}(\mathbf{n}))^2h^4 - 2f_{n-2}(\mathbf{n})\left(f_n(\mathbf{n}) + D_\Delta(\mathbf{n}) \cdot f_{n-2}(\mathbf{n})\right)h^2 +& \\ + \left(f_n(\mathbf{n}) - D_\Delta(\mathbf{n}) \cdot f_{n-2}(\mathbf{n})\right)^2&.\end{aligned} \tag{32}$$

Next, trying to rewrite (32) as a product of two polynomials quadratic in h we have to guarantee that the discriminant is a perfect square, which is equivalent to the condition

$$D_\Delta(\mathbf{n}) \cdot f_{n-2}(\mathbf{n}) \cdot f_n(\mathbf{n}) = \sigma^2(\mathbf{n}). \tag{33}$$

Finally, under the condition on $F^\vee(\mathbf{n}, h)$ to be irreducible we arrive at (31). □

Table 1. Some examples of fundamental QN curves. The three columns correspond to the dual equations $D_\Delta f^2h^2 + g^4 = 0$, $f^2h^2 + D_\Delta g^2 = 0$ and $\delta_1 f^2h^2 + \delta_2 g^3 = 0$, respectively, where $g = n_1$ and δ_i are nonconstant factors of D_Δ. Cross denotes a non-real fundamental QN curve

D_Δ	f	$(D_\Delta, 1)$	$(1, D_\Delta)$	(δ_1, δ_2)
$n_1^2 + n_2^2$	n_2			×
	$n_1 + 2n_2$			×
$n_1^2 - n_2^2$	n_2			
	$n_1 + 2n_2$			

In addition, similarly to the case of LN curves we may use the partial fraction decomposition to obtain

$$h = \sqrt{-\frac{d_2(\mathbf{n})}{d_1(\mathbf{n})}} \cdot \frac{g(\mathbf{n})}{f(\mathbf{n})} = h_0 + \cdots + h_\ell, \tag{34}$$

where

$$h_i = \sqrt{-\frac{d_2(\mathbf{n})}{d_1(\mathbf{n})}} \cdot \frac{\alpha_i n_1^{k+d}}{(\beta_i n_1 + \gamma_i n_2)^k} \tag{35}$$

for some $\alpha_i, \beta_i, \gamma_i \in \mathbb{C}$, $k \in \mathbb{N}$, and $d = 0$ if $d_1 = D_\Delta$, $d = 2$ if $d_2 = D_\Delta$ and $d = 1$ otherwise. Hence we arrive at

Proposition 3. *Any centrally symmetric QN curve $\mathcal{V}$ can be obtained as a component of the convolution $\mathcal{Q}_1 \star \cdots \star \mathcal{Q}_\ell$ where the associated dual equations have the form*

$$\mathcal{Q}_i^\vee : d_1(\mathbf{n})(\beta_i n_1 + \gamma_i n_2)^{2k} h^2 + d_2(\mathbf{n}) \alpha_i^2 n_1^{2(k+d)}. \tag{36}$$

Remark 7. Surprisingly, we do not need all four types of the dual equations determined by the decomposition $D_\Delta = d_1 \cdot d_2$. It may be shown that any curve in $\mathfrak{Q}^0_\Delta$ can be either written dually in the form $D_\Delta(\mathbf{n})f^2(\mathbf{n})h^2 + g^2(\mathbf{n}) = 0$, or it is a component of the convolution of two such curves – the decomposition can be consequently applied only to this class.

5 Conclusion

In this paper we continued the study from [1] and extended the ideas concerning curves with low convolution degree. As a new result, we have proved that the so called LN curves (curves with convolution degree one) can be obtained as the convolution of a finite number of canonical curves. The main contribution of the paper is in Section 4, where curves of convolution degree two are investigated. We presented their elementary properties including the fact that these curves are either rational, elliptic or hyper-elliptic. Next, the notion of QN curves (rational curves with convolution degree two) was introduced and properties of these curves were thoroughly studied. Finally, a canonical decomposition of an arbitrary QN curve was presented.

We believe that the curves with low convolution degree are objects interesting not only from the purely geometric point of view but also as curves which can serve as primitives for approximate convolution process. These curves comprise all conic sections (LN parabolas, and QN ellipses and hyperbolas) and therefore all interpolation/approximation algorithms based on (arcs of) conic sections can be immediately used as the first step. In addition, all (M)PH interpolation/approximation techniques may be easily modified and then used.

Acknowledgments. The second author was supported by the Research Plan MSM 4977751301. We thank all referees for their comments, which helped us to improve the paper.

References

1. Vršek, J., Lávička, M.: On convolution of algebraic curves. Journal of Symbolic Computation 45(6), 657–676 (2010)
2. Farouki, R.: Pythagorean-Hodograph Curves: Algebra and Geometry Inseparable. Springer, Heidelberg (2008)
3. Farouki, R., Sakkalis, T.: Pythagorean hodographs. IBM Journal of Research and Development 34(5), 736–752 (1990)
4. Farouki, R., Sakkalis, T.: Pythagorean-hodograph space curves. Adv. Comput. Math. 2, 41–66 (1994)
5. Kosinka, J., Jüttler, B.: G^1 Hermite interpolation by Minkowski Pythagorean hodograph cubics. Computer Aided Geometric Design 23, 401–418 (2006)
6. Kosinka, J., Jüttler, B.: MOS surfaces: Medial surface transforms with rational domain boundaries. In: Martin, R., Sabin, M.A., Winkler, J.R. (eds.) Mathematics of Surfaces 2007. LNCS, vol. 4647, pp. 245–262. Springer, Heidelberg (2007)
7. Kosinka, J., Lávička, M.: On rational Minkowski Pythagorean hodograph curves. Computer Aided Geometric Design 27(7), 514–524 (2010)
8. Peternell, M., Pottmann, H.: A Laguerre geometric approach to rational offsets. Computer Aided Geometric Design 15, 223–249 (1998)

9. Pottmann, H., Peternell, M.: Applications of Laguerre geometry in CAGD. Computer Aided Geometric Design 15, 165–186 (1998)
10. Jüttler, B.: Triangular Bézier surface patches with linear normal vector field. In: Cripps, R. (ed.) The Mathematics of Surfaces VIII. Information Geometers, pp. 431–446 (1998)
11. Peternell, M., Manhart, F.: The convolution of a paraboloid and a parametrized surface. Journal for Geometry and Graphics 7(2), 157–171 (2003)
12. Sampoli, M.L., Peternell, M., Jüttler, B.: Rational surfaces with linear normals and their convolutions with rational surfaces. Computer Aided Geometric Design 23(2), 179–192 (2006)
13. Lávička, M., Bastl, B.: Rational hypersurfaces with rational convolutions. Computer Aided Geometric Design 24(7), 410–426 (2007)
14. Lee, I.K., Kim, M.S., Elber, G.: Polynomial/rational approximation of Minkowski sum boundary curves. Graphical Models and Image Processing 60(2), 136–165 (1998)
15. Gravesen, J., Jüttler, B., Šír, Z.: On rationally supported surfaces. Computer Aided Geometric Design 25, 320–331 (2008)
16. Šír, Z., Gravesen, J., Jüttler, B.: Curves and surfaces represented by polynomial support functions. Theoretical Computer Science 392(1-3), 141–157 (2008)
17. Arrondo, E., Sendra, J., Sendra, J.R.: Parametric generalized offsets to hypersurfaces. Journal of Symbolic Computation 23, 267–285 (1997)
18. Arrondo, E., Sendra, J., Sendra, J.R.: Genus formula for generalized offset curves. Journal of Pure and Applied Algebra 136, 199–209 (1999)
19. Sendra, J.R., Sendra, J.: Algebraic analysis of offsets to hypersurfaces. Mathematische Zeitschrift 237, 697–719 (2000)
20. Šír, Z., Bastl, B., Lávička, M.: Hermite interpolation by hypocycloids and epicycloids with rational offsets. Computer Aided Geometric Design 27, 405–417 (2010)
21. Jüttler, B.: Hermite interpolation by Pythagorean hodograph curves of degree seven. Math. Comp. 70, 1089–1111 (2001)
22. Jüttler, B., Sampoli, M.: Hermite interpolation by piecewise polynomial surfaces with rational offsets. Computer Aided Geometric Design 17, 361–385 (2000)
23. Meek, D.S., Walton, D.J.: Geometric Hermite interpolation with Tschirnhausen cubics. J. Comput. Appl. Math. 81(2), 299–309 (1997)
24. Brieskorn, E., Knörer, H.: Plane algebraic curves. Birkhaüser, Basel (1986)
25. Cox, D.A., Little, J., O'Shea, D.: Using algebraic geometry, 2nd edn. Springer, Heidelberg (2005)
26. Fulton, W.: Algebraic Curves. Benjamin, New York (1969)
27. Kim, M.S., Elber, G.: Problem reduction to parameter space. In: Proceedings of the 9th IMA Conference on the Mathematics of Surfaces, pp. 82–98. Springer, Heidelberg (2000)
28. Walker, R.: Algebraic Curves. Princeton University Press, Princeton (1950)
29. Lávička, M., Bastl, B., Šír, Z.: Reparameterization of curves and surfaces with respect to their convolution. In: Dæhlen, M., Floater, M., Lyche, T., Merrien, J.-L., Mørken, K., Schumaker, L.L. (eds.) MMCS 2008. LNCS, vol. 5862, pp. 285–298. Springer, Heidelberg (2010)
30. Cohen, H., Frey, G. (eds.): Handbook of Elliptic and Hyperelliptic Curve Cryptography. Chapman & Hall/CRC (2005)
31. Hartshorne, R.: Algebraic Geometry. Springer, Heidelberg (1977)
32. Moon, H.: Minkowski Pythagorean hodographs. Computer Aided Geometric Design 16, 739–753 (1999)

A Logistic Model for the Degradation of Triangle Mesh Normals

Ying Yang and Ioannis Ivrissimtzis

School of Engineering and Computing Sciences, Durham University, UK

Abstract. The performance of shading and ray-tracing algorithms depends heavily on the quality of the surface normal information. As a result, in many visual applications normal information turns out to be more important than spatial information. This paper proposes a logistic model for the degradation of the normal information resulting from the quantisation of the vertex coordinates. The mesh is degraded by the randomization of each vertex coordinate after its t-th significant bit. The normal degradation is computed as a weighted average of the angle differences between the normals of the original triangles and the corresponding degraded triangles. The proposed model is validated experimentally. As an application, we use the proposed logistic model to estimate suitable levels of quantisation for 3D triangle meshes.

Keywords: triangle mesh, model degradation, logistic models.

1 Introduction

In graphics and visualisation applications, triangle mesh is the ubiquitous standard for surface representation. Triangle meshes also play an important role in CAD/CAM applications when scans of physical objects are processed into triangle mesh models, or when NURBS surfaces are converted into triangle meshes for fast interactive visualisations.

A triangle mesh approximates a surface embedded in 3D by a set of 3D points, called *vertices*, connected between them with a set of triangles called *faces*. Each vertex is described in an orthonormal Cartesian system by its x, y, z coordinates. As any digital information, the vertex coordinates can be seen as real numbers quantised at a level l, with typical values $l = 32$ bits (floats), or $l = 64$ bits (doubles). Regarding the quantisation method, here we assume that any coordinate bit after the chosen level of quantisation l has a random value, rather than being zero.

The choice of the appropriate l is a trade-off between efficiency and quality. A small l may lead to a significant loss of geometric information while, on the other hand, a large l may lead to mesh representations with a lot of redundancy, resulting to unnecessarily large files and consequently, to unnecessarily expensive computations.

This paper proposes a simple mathematical model for the trade-off between encoding efficiency and mesh quality. More specifically, it proposes a logistic model to predict the degradation of the face normals as the level of quantisation decreases. We notice that modelling normal degradation can be extremely useful in practical applications because the quality of the mesh renderings is more dependent on the quality of normal information than the quality of spatial information. The reason is that most rendering algorithms for triangle meshes use normals to do the lighting computations and eventually determine the colour of each pixel on the viewer's screen.

As an application of the proposed normal degradation model, we will compute appropriate levels of quantisation, given a tolerance for the average accuracy of a triangle normal, possibly weighted by geometric characteristics of the triangle, such as its area, or its dihedral angles. Finally, we will demonstrate by several examples that the claimed optimisation is visually meaningful.

1.1 Previous Work

Most rendering algorithms, from Gouraud [1] and Phong [2] shading to the computationally intensive BRDF techniques [3], make use of face or vertex normal information. Vertex normals are usually computed as weighted averages of face normals [4], and a similar degradation behaviour is expected. In some applications, normals are not computed using spatial mesh information, but are separately obtained at the data acquisition stage and encoded as 3D vectors by three 32 bit coordinates. [5] studies the quantisation error introduced by such normal representations and proposes efficient vector encoding methods.

The randomisation of the least significant bits adds a stochastic element to the mesh geometry. Uncertainty in polygonal meshes has been studied in [6,7]. The retention of the most significant bits of a signal and the substitution of the least significant bits by random bits is a commonly used technique called *dithering* [8]. Dithering is used extensively in digital signal processing to prevent quantisation artifacts, that is, unwanted regular patterns that may distract the eye or the ear.

The problem of finding the appropriate level of mesh quantisation has been encountered in mesh compression. In particular, the appropriate selection of quantisation level is necessary for the accuracy of the parallelogram prediction rule [9,10], which is still considered the state-of-the-art of predictive mesh encoding [11]. The importance of removing redundant bits through vertex quantisation is further magnified by the fact that, typically, the predictions of the least significant bits have high entropy and thus, they cannot be efficiently compressed [12]. However, despite its importance, the problem of choosing appropriate levels of quantisation has not been systematically studied and all proposed methods rely on the intuition of the user.

1.2 Contribution and Limitations

The main contributions of the paper are:

- A logistic model describing the degradation of the normal information of a triangle mesh as the quantisation level decreases.

– A method for computing an appropriate level of mesh quantisation when a tolerance for the accuracy of the normals is given.

The main limitation of our approach is the assumption that our meshes have no significant amount of noise. We make this assumption implicitly, by regarding the normals at the highest level of quantisation as the most accurate. Thus, even though geometric noise can be estimated [13], and the effects of noise at different levels of mesh quantisation have been empirically studied [14], this paper focuses on the effects of quantisation on clean, high quality meshes.

1.3 Overview

The rest of the paper is organised as follows. In Section 2, we describe a logistic model for normal degradation and show how it can be empirically computed for a given triangle mesh. In Section 3, we experimentally validate the proposed model. In Section 4, we use the degradation model to compute appropriate quantisations for triangle meshes and experimentally show that the claimed optimisation is visually meaningful. We briefly conclude in Section 5.

2 A Logistic Model for Normal Degradation

Given a triangle mesh $\mathcal{M}$, we model the quality of its normal information as a function of the level of quantisation t by

$$\mathsf{D}_{\mathcal{M}}(t) = C/(1 + e^{-a-bt}), \quad t \geq 0. \tag{1}$$

$\mathsf{D}_{\mathcal{M}}$ is the expected average change of the triangle normals, possibly weighted by geometric characteristics of the triangles, when the t most significant bits of each vertex coordinate are retained and the less significant bits are randomised. Under our assumption of a clean, high quality original mesh $\mathcal{M}$, $\mathsf{D}_{\mathcal{M}}$ is seen as the normal error resulting from the vertex quantisation. We notice that we can see (1) as an abstract degradation model and think of t as a real number; however, in practice, t represents bit positions and thus we are interested in the integer values of t between 0 and 64.

The curve of (1) has an inverse 'S' shape. Small values of t correspond to coarse quantisations and large expected normal error, while large values of t correspond to fine quantisations and small expected normal error. The exact shape of the curve depends on the three constants a, b and C. The value of a represents a quantisation threshold, after which some of the normal information of the original mesh is retained, and b represents the rate at which normal information is retained. C is a scaling factor controlling the maximum value of $\mathsf{D}_{\mathcal{M}}$. We notice that the maximum of $\mathsf{D}_{\mathcal{M}}$ should be obtained at $t = 0$, that is, when all spatial information is random, in which case the expected average normal error should reach its theoretical maximum of $\pi/2$. For the usual range of values of a and $t = 0$, we have found experimentally that $(1 + e^{-a-bt}) \approx 1$ and thus $C \approx \pi/2$.

Equation (1) is a member of the family of the *logistic functions*. These functions were initially introduced to model population growth, and have since found numerous applications in fields ranging from social sciences to engineering. In some of these applications, logistic functions have been used as degradation models, describing for example the transition of the state of a machine from working perfectly to total failure [15].

2.1 Logistic Curve Fitting

For a given mesh $\mathcal{M}$, its degradation model, that is, the values of the constants a, b and C, is computed empirically. Specifically, for several integer values of t between 0 and 64, we randomise all vertex coordinates after their t-th bit and compute the weighted average change of the face normals. This weighted average is considered a sample from the logistic curve at parameter value t. The curve itself is computed by applying logistic curve fitting on samples computed at several values of t.

To describe the above fitting process more formally, let the i-th vertex of a given triangle mesh $\mathcal{M}$ be given in Cartesian coordinates by $v_i = (x_i, y_i, z_i)$ $(1 \leq i \leq N, i \in \mathbb{N})$, where N is the number of vertices of $\mathcal{M}$. As the coordinates are assumed to be float-point numbers, they can be represented in double precision format (64-bit long). After this format conversion the mesh vertices have the form $\hat{v}_i = (\hat{x}_i, \hat{y}_i, \hat{z}_i)$, where $\hat{x}_i, \hat{y}_i$ and $\hat{z}_i$ are binary strings of 64 bits.

Next, we randomise the least significant bits. Specifically, for each $\hat{v}_i = (\hat{x}_i, \hat{y}_i, \hat{z}_i)$, we retain the t $(0 \leq t \leq 64, t \in \mathbb{N})$ most significant bits of each of $\hat{x}_i$, $\hat{y}_i$ and $\hat{z}_i$ and replace the $64 - t$ least significant bits with randomly generated bits. The result is a new set of coordinates $\check{v}_i = (\check{x}_i, \check{y}_i, \check{z}_i)$, which are converted into the floating point coordinates $v'_i = (x'_i, y'_i, z'_i)$ of the degraded mesh $\mathcal{M}^t$.

Next, we compare the triangle normals of $\mathcal{M}$ and $\mathcal{M}^t$ and the normal degradation is measured as the weighted average of the normal distortion

$$\mathsf{dis}(\mathcal{M}, \mathcal{M}^t) = \frac{\sum_{i=1}^{M} w_i \cdot \mathsf{angle}(\hat{\mathbf{n}}_i, \hat{\mathbf{n}}_i^t)}{\sum_{i=1}^{M} w_i}, \tag{2}$$

where M is the number of triangles in $\mathcal{M}$, $\hat{\mathbf{n}}_i$ and $\hat{\mathbf{n}}_i^t$, are the normals of the i-th triangle of $\mathcal{M}$ and $\mathcal{M}^t$, respectively, $\mathsf{angle}(\hat{\mathbf{n}}_i, \hat{\mathbf{n}}_i^t)$ is the smaller angle between $\hat{\mathbf{n}}_i$ and $\hat{\mathbf{n}}_i^t$ expressed in radians.

If we put $w_i = 1$ for $i = 1, 2, \ldots, M$ we get the mean average of the normal distortion. Depending on the application, we might want to weight the average in (2) by the area A_i of the triangles, that is, $w_i = A_i$ for $i = 1, 2, \ldots, M$. In this case, the normal distortion of the larger triangles, which dominate the rendering process, has a larger weight. A third possibility is to use larger weights for triangles with small dihedral angles. Such triangles represent the flat areas of the surface where even very small normal distortions can be immediately perceived as noise. For a dihedral angle weighted average we used the Gaussian weights

$$w_i = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\dfrac{(x_i-\mu)^2}{2\sigma^2}} \tag{3}$$

where x_i is the smallest of the three dihedral angles of the i-th triangle. In the experiments, we fixed $\mu = 0$ and $\sigma = 3.5$, which gave reasonable results.

Regarding the properties of $\mathsf{dis}(\mathcal{M}, \mathcal{M}^t)$, from

$$0 \leq \mathsf{angle}(\hat{\mathbf{n}}_i, \hat{\mathbf{n}}_i^t) \leq \pi, \quad 0 \leq t \leq 64, \tag{4}$$

we get

$$0 \leq \mathsf{dis}(\mathcal{M}, \mathcal{M}^t) \leq \pi, \quad 0 \leq t \leq 64. \tag{5}$$

While it seems quite difficult to improve these deterministic bounds, nevertheless, regarding the expectations for $\mathsf{dis}(\mathcal{M}, \mathcal{M}^t)$, we can easily see that the expectation $E(\mathsf{dis}(\mathcal{M}, \mathcal{M}^t))$ is a decreasing function of t and that $E(\mathsf{dis}(\mathcal{M}, \mathcal{M}^0)) \approx \pi/2$ as already discussed. Moreover, all our experiments with commonly used triangle meshes confirm that $\mathsf{dis}(\mathcal{M}, \mathcal{M}^t) \approx 0$ as t approaches 64.

Finally, the last step of the process is to use logistic curve fitting and fit the logistic model of (1) to the samples computed by (2).

2.2 Estimation of Appropriate Quantisation Levels

The proposed logistic model can be used to find the appropriate level of quantisation when a tolerance for the expected normal error is given. Indeed, by solving (1) we get

$$t = -\Big(\ln \frac{C - \mathsf{D}(\mathcal{M}, \mathcal{M}^t)}{\mathsf{D}(\mathcal{M}, \mathcal{M}^t)} + a\Big)/b \tag{6}$$

The parameters a, b and C are experimentally computed as described in Section 2.1. Then, we substitute the normal error predicted by the model, i.e. $\mathsf{D}(\mathcal{M}, \mathcal{M}^t)$, with the given tolerance and compute the appropriate level of quantization t from (6).

Equation (6) can be further simplified by assuming $C = \pi/2$ and a fixed tolerance that would be acceptable for all intended applications. For example, for a normal error tolerance of $\epsilon = 1^\circ$ (≈ 0.01745 radians), which is acceptable in most visualisation applications, after using the ceiling function to convert t to an integer, (6) becomes

$$t = \lceil -(4.489 + a)/b \rceil \tag{7}$$

Thus, using the proposed logistic model, we are able to figure out the suitable quantisation level easily, once the normal error $\mathsf{D}(\mathcal{M}, \mathcal{M}^t)$ and the two parameters a and b are given.

Table 1. Mesh details and the results of the logistic model fitting. For each of the mean average $\mathsf{dis}^{av}(\mathcal{M}, \mathcal{M}^t)$, area weighted average $\mathsf{dis}^{ar}(\mathcal{M}, \mathcal{M}^t)$ and dihedral angle weighted average $\mathsf{dis}^{an}(\mathcal{M}, \mathcal{M}^t)$, the **left**, **middle** and **right** columns show the values of a, b and C, respectively.

	M	$\mathsf{dis}^{av}(\mathcal{M}, \mathcal{M}^t)$			$\mathsf{dis}^{ar}(\mathcal{M}, \mathcal{M}^t)$			$\mathsf{dis}^{an}(\mathcal{M}, \mathcal{M}^t)$		
Fandisk	12946	22.863	-1.168	1.573	23.108	-1.184	1.573	22.864	-1.168	1.573
Bunny	69666	22.300	-1.154	1.570	22.127	-1.148	1.570	22.315	-1.153	1.570
Dragon	100000	18.865	-0.927	1.571	20.874	-1.077	1.572	19.095	-0.939	1.569
Lucy	525814	23.503	-1.081	1.571	21.887	-1.028	1.571	23.307	-1.071	1.572
MPII Geometry	70761	14.628	-0.685	1.569	11.600	-0.703	1.568	14.624	-0.684	1.571

3 Validation and Quantisation Levels

In this section we experimentally validate the proposed logistic model and compute the appropriate levels of quantisation based on this model. The former is to test the accuracy of the model, that is, to measure the difference between the prediction of the model $\mathsf{D}(\mathcal{M}, \mathcal{M}^t)$ and the observed average normal distortion $\mathsf{dis}(\mathcal{M}, \mathcal{M}^t)$. By contrast, the latter is to find out the appropriate level of quantisation, subject to a given normal degradation.

The validation experiment was conducted on a set of synthetic and natural 3D triangle mesh models consisting of the *Fandisk*, *Bunny*, *Dragon*, *Lucy* and *MPII Geometry*. The mesh details and the results of the fitting process are summarised in Table 1. We notice that in all cases the value of the threshold a is relatively large, meaning that 8-bit or even 12-bit mesh vertex quantisations result to the loss of almost all normal information. This is in contrast to the resiliency of the volumetric properties of the corresponding shapes, given that in 12-bit, or even 8-bit voxelisations, the shapes are still clearly recognisable. We also notice that the large value of a means that $C \approx \pi/2$ for each of the test meshes, as expected.

The comparisons between the predictions of the logistic model $\mathsf{D}(\mathcal{M}, \mathcal{M}^t)$ and the corresponding experimental observations $\mathsf{dis}(\mathcal{M}, \mathcal{M}^t)$ are shown in Fig.1. The logistic model fits five data points computed at $t = 12$, 16, 20, 24 and 28 (see (2)). From Fig.1, we see that the red and blue curves of $\mathsf{D}(\mathcal{M}, \mathcal{M}^t)$ and $\mathsf{dis}(\mathcal{M}, \mathcal{M}^t)$, respectively, are almost identical. That means that the proposed logistic model can successfully predict the quality of the triangle normals. The difference between the two curves are generally small, except at the high slope part of the curves. We notice that the higher error at the high slope part of the curve was predictable, given that small misalignments at the horizontal direction can cause large discrepancies at the vertical direction. We also notice that quantisations corresponding to that part of the curve are not of direct interest in practical applications because their normal error is very large.

Fig. 1 shows that the logistic functions fit nicely the data from the five test models, except for the MPII Geometry model for some values of t for the case of area weighted average. The problem with the MPII Geometry model is caused by the high variance of the normal degradation, which as a result cannot be

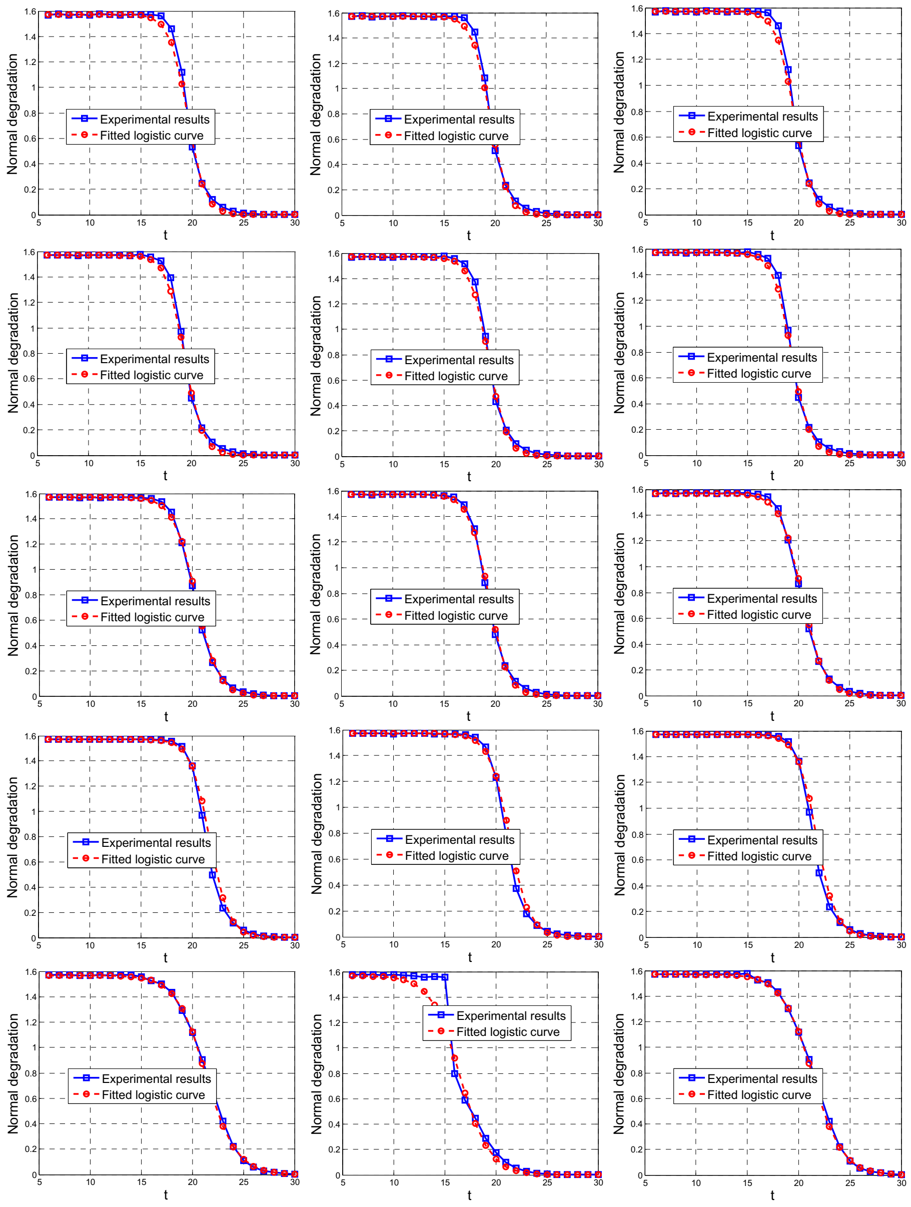


Fig. 1. Left: Comparison between the average normal distortion observations $\mathsf{dis}^{av}(\mathcal{M}, \mathcal{M}^t)$ and the logistic model predictions. **Middle:** Comparison between the area-weighted average of normal distortion observations $\mathsf{dis}^{ar}(\mathcal{M}, \mathcal{M}^t)$ and the logistic model predictions. **Right:** Comparison between the dihedral angle-weighted average of normal distortion observations $\mathsf{dis}^{an}(\mathcal{M}, \mathcal{M}^t)$ and the logistic model predictions. The models used here are *Fandisk*, *Bunny Dragon*, *Lucy* and *MPII Geometry* (from top to bottom).

Table 2. Appropriate quantization levels. For each of the mean average $\mathsf{dis}^{av}(\mathcal{M}, \mathcal{M}^t)$, area weighted average $\mathsf{dis}^{ar}(\mathcal{M}, \mathcal{M}^t)$ and dihedral angle weighted average $\mathsf{dis}^{an}(\mathcal{M}, \mathcal{M}^t)$, the **left**, **middle** and **right** columns correspond to $\epsilon = 1°$, $\epsilon = 5°$ and $\epsilon = 10°$, respectively

	$\mathsf{dis}^{av}(\mathcal{M}, \mathcal{M}^t)$			$\mathsf{dis}^{ar}(\mathcal{M}, \mathcal{M}^t)$			$\mathsf{dis}^{an}(\mathcal{M}, \mathcal{M}^t)$		
Fandisk	24	22	22	24	22	22	24	22	22
Bunny	24	22	22	24	22	22	24	22	22
Dragon	26	24	23	24	22	22	26	24	23
Lucy	26	25	24	26	25	24	26	25	24
MPII Geometry	28	26	25	23	21	20	28	26	25

modelled effectively. In particular, for some values of t, the normal degradation variance of the very large triangles is also very large and dominates the area weighted average. We notice that the MPII model is the only one of our test models with very large triangles, as it contains triangles spanning whole sides of the building. Fig. 2 shows the variance computations for the test models.

As mentioned before, the proposed model is quite efficient in estimating the suitable mesh quantisation level, given a normal degradation. Indeed, the appropriate levels of quantisation t can be yielded by simply rounding t in (6) to the nearest integers towards infinity. For example, for $\epsilon = 1°$, the levels t for the five models of the validation experiment, computed according to (7), are shown in Table 2. The appropriate levels of quantisation for $\epsilon = 5°$ and $10°$ can be computed analogously. We notice that, as expected, different models may have different appropriate level of quantisation. For example, when the tolerance is $\epsilon = 1°$ and we consider the mean average of the normal distortion, the relatively simple *Fandisk* model can be represented by 24 bits per vertex coordinate without significant loss of normal information, while the more complex *Lucy* and *MPII Geometry* models require 26 and 28 per vertex coordinate, respectively. We also notice that, in most models, the appropriate level of quantisation does not depend on the chosen averaging method. A notable exception is the *MPII Geometry* which consists of some extremely large and some extremely small triangles and the results of the area weighted averaging method diverge.

To judge the visual significance of the claimed optimisation of t, the left column of Fig. 3 shows renderings of the original meshes quantised at 64 bits per vertex coordinate and the middle column renderings of their respective appropriate quantisations for a small tolerance of $\epsilon \approx 1°$ when the mean averaging method is used. We notice that in all cases the appropriately quantised meshes are almost indistinguishable from the originals, and any possible degradation has been kept to visually acceptable levels.

To further demonstrate the relevance of our results, the right column of Fig. 3 shows renderings of the same meshes quantised at a coarser level corresponding to a tolerance of $\epsilon \approx 10°$. In practice that means that they are quantised at a level that is 2 or 3 bits coarser than the ones on the middle column. We notice

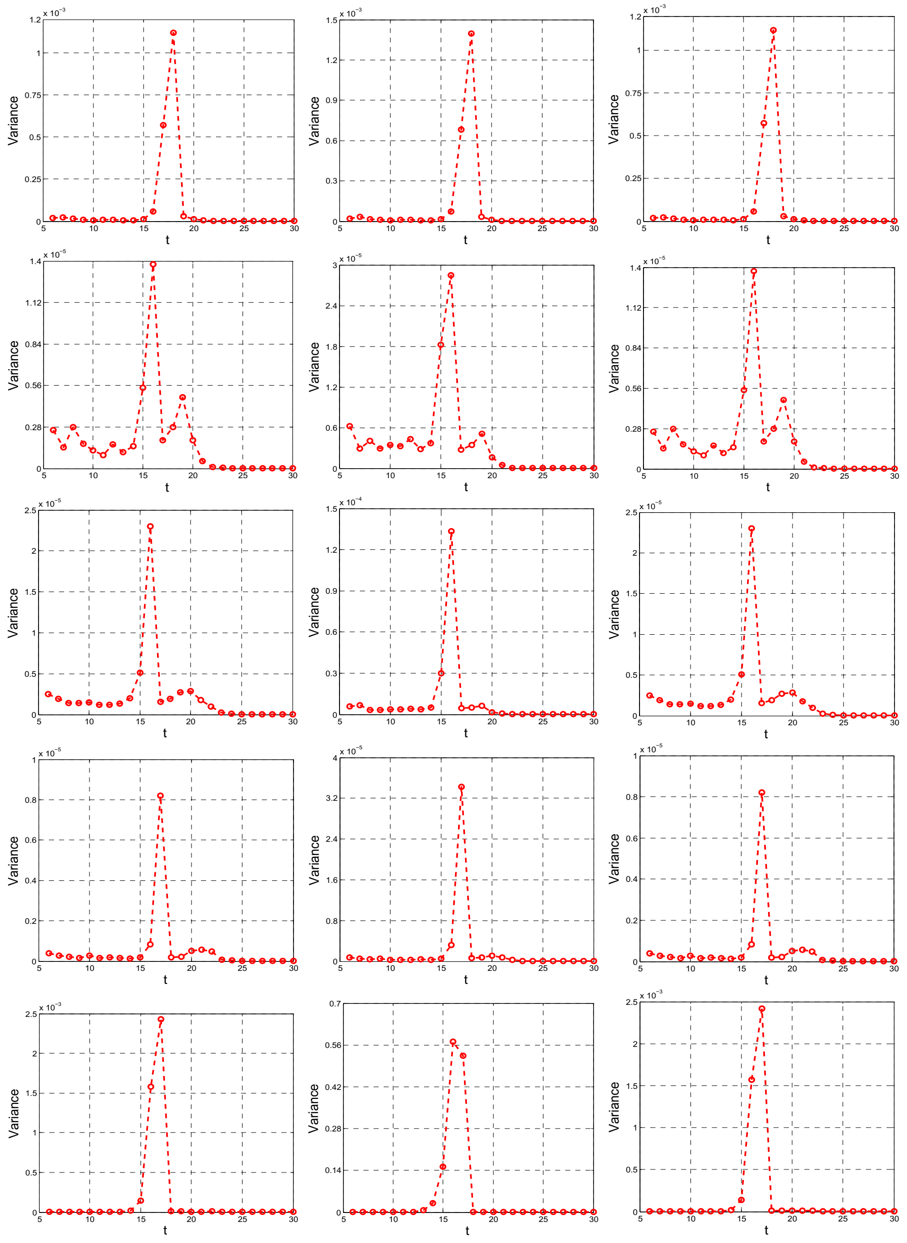


Fig. 2. Left: Variances of the average normal distortion observations $\mathsf{dis}^{av}(\mathcal{M}, \mathcal{M}^t)$. **Middle:** Variances of the area-weighted average of normal distortion observations $\mathsf{dis}^{ar}(\mathcal{M}, \mathcal{M}^t)$. **Right:** Variances of the dihedral angle-weighted average of normal distortion observations $\mathsf{dis}^{an}(\mathcal{M}, \mathcal{M}^t)$. The models used here are *Fandisk*, *Bunny Dragon*, *Lucy* and *MPII Geometry* (from top to bottom).

that in all cases except the *Dragon* the degradation of the normal information is visually significant. The only exception is the *Dragon* model which does not show significant signs of degradation as a result of the coarser than appropriate quantisation. The reason is that the original *Dragon* model is already noisy and thus, its original normal information has already a large random element which is not affected by the quantisation. This salient point is further illustrated in Fig. 4 where close ups of the *Fandisk* and the *Bunny* at appropriate and suboptimal levels of quantisation are shown. Finally, we notice that a given level of quantisation ($t = 24$) may be appropriate for one mesh (*Fandisk*) and significantly suboptimal for another (*Lucy*). Fig. 4 shows close-ups of the models in Fig. 3.

4 Practical Applications

In this section we briefly discuss how the proposed logistic model for mesh degradation can be used to enhance existing mesh steganography/watermarking and mesh compression algorithms, help to evaluate such algorithms and inform their further development.

The goal of steganography is to embed a confidential message on a carrier signal, here a mesh model, in such a way that no one apart from the sender and the intended recipient can detect the existence of the hidden message. The challenge in designing a good steganographic algorithm lies in balancing the two conflicting requirements of high *embedding capacity* and low *embedding distortion*. That is, one has to maximise the length of the message that can be embedded on a given mesh and simultaneously minimise the visual impact of that embedding.

We notice that the proposed logistic model describes a trade-off between embedding capacity, in the form of unused bits in the vertex coordinates, and distortion, in the form of normal degradation. Therefore, it can directly be used to inform least significant bit steganographic algorithms about the embedding capacity of the carrier. More specifically, for a given carrier mesh $\mathcal{M}$, we first compute the parameters a, b of the logistic degradation model, as described in Section 2. Then, for a certain distortion tolerance $\mathsf{D}(\mathcal{M}, \mathcal{M}^t)$ we compute the appropriate level of quantisation t using (6). Finally, the $64 - t$ least significant bits of each vertex coordinate are replaced with the message bits to be hidden.

The proposed logistic model can also be used in conjunction with mesh compression algorithms, as discussed in the Introduction. In particular, it can be used to inform the user's choice of level of quantisation t for the vertex coordinates. After t has been determined, the compression algorithm will only keep the t most significant bits of each coordinate and the coarse mesh will be compressed and transmitted. At the receiver end, the coarse mesh will be decoded and $64-t$ random bits will be appended to the t most significant bits of each coordinate. Such a process would be completely analogous to dithering, as used in signal encoding and compression.

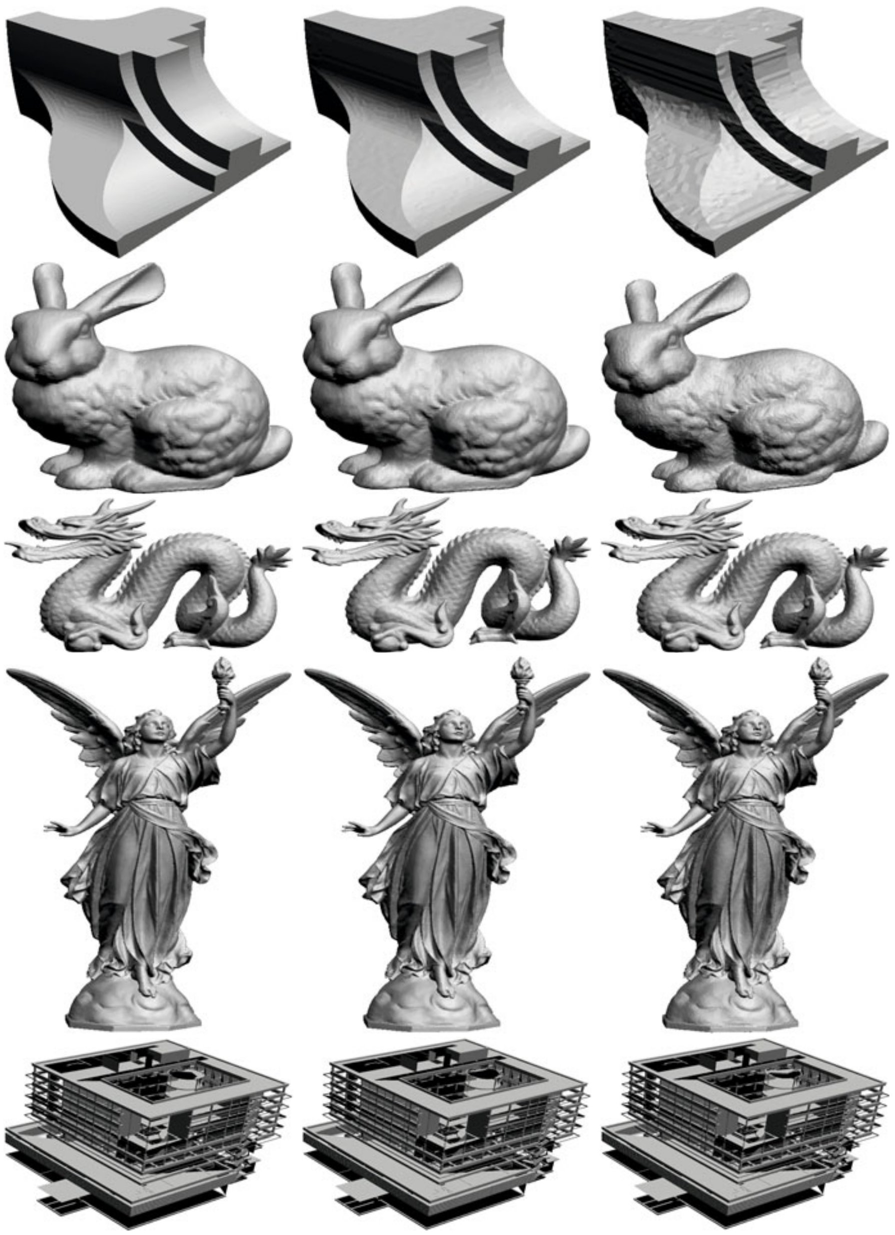


Fig. 3. Left: The original *Fandisk*, *Bunny*, *Dragon*, *Lucy* and *MPII Geometry* meshes quantised at 64 bits per vertex coordinate. **Middle:** Finely quantised meshes at 24, 24, 26, 26 and 28 bits per vertex coordinate, respectively ($\epsilon \approx 1^\circ$). **Right:** Coarsely quantised meshes at 22, 22, 23, 24 and 25 bits per vertex coordinate, respectively ($\epsilon \approx 10^\circ$).

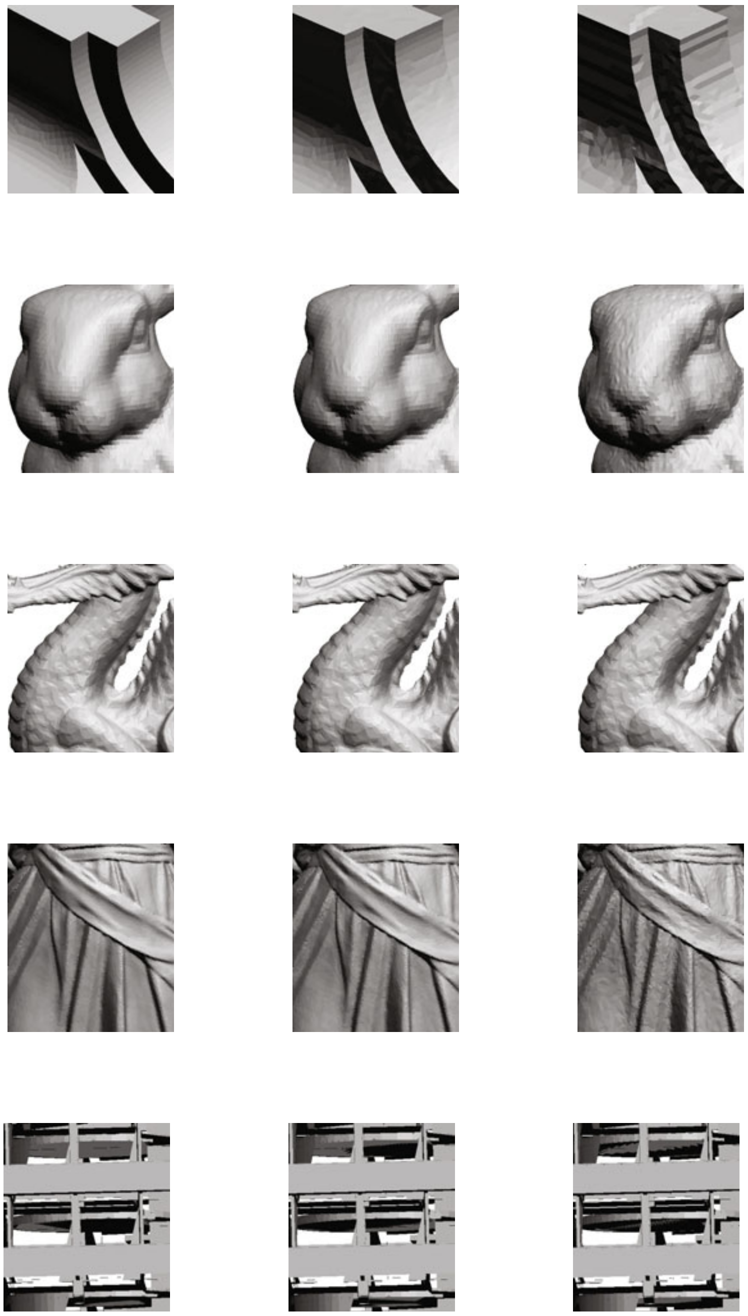


Fig. 4. Close-ups of the models in Fig. 3

5 Conclusions and Future Work

We proposed a logistic model for estimating the degradation of face normals in 3D triangle meshes caused the quantisation of vertex coordinates. As demonstrated by the validation experiments, the behaviour of the proposed model is satisfactory and its predictions for the normal degradation are good approximations of the respective experimental values. Finally, we discussed how the proposed model might be utilised in a number of applications, especially those requiring the estimation of an appropriate quantisation level for vertex coordinates, as for example mesh steganography/watermarking and mesh compression.

In the future, we plan to use the logistic degradation model to study existing high capacity mesh steganographic algorithms [16] and their counterpart robust mesh watermarking algorithms [17]. In particular, the algorithms will be tested and their evaluation will take into account the resilience of the carrier meshes against distortion, as encapsulated in the parameters a and b of the logistic model. We believe that this approach can open the way to evaluation methodologies for steganographic/watermarking algorithms that will be more rigorous than the current common practice.

Another direction of our future research is to develop mathematical formula estimating the parameters a and b from simple mesh characteristics related to the number, size and shape of the triangles. Such formula will allow the direct computation of the logistic model, with no need for a computationally expensive simulation of mesh degradation, as described in Section 2, to collect data for model fitting.

References

1. Gouraud, H.: Continuous Shading of Curved Surfaces. IEEE Transactions on Computers 20, 623–629 (1971)
2. Phong, B.T.: Illumination for Computer Generated Pictures. ACM Commun. 18, 311–317 (1975)
3. Walter, B., Zhao, S., Holzschuch, N., Bala, K.: Single Scattering in Refractive Media with Triangle Mesh Boundaries. In: SIGGRAPH, pp. 1–8. ACM Press, New York (2009)
4. Jin, S., Lewis, R.R., West, D.: A Comparison of Algorithms for Vertex Normal Computation. The Visual Computer 21, 71–82 (2005)
5. Meyer, Q., Susharpmuth, J., Susharpner, G., Stamminger, M., Greiner, G.: On Floating-Point Normal Vectors. Computer Graphics Forum 29, 1405–1409 (2010)
6. Pauly, M., Mitra, N.J., Guibas, L.J.: Uncertainty and Variability in Point Cloud Surface Data. In: Symposium on Point-Based Graphics, pp. 77–84. Eurographics Press, Aire-la-Ville (2004)
7. Kalaiah, A., Varshney, A.: Statistical Geometry Representation for Efficient Transmission and Rendering. ACM Transactions on Graphics 24, 348–373 (2005)
8. Gray, R., Stockham, T.: Dithered quantizers. IEEE Transactions on Information Theory 39, 805–812 (1993)
9. Touma, C., Gotsman, C.: Triangle Mesh Compression. In: Graphics Interface, pp. 26–34. Canadian Human-Computer Communications Society, Waterloo (1998)

10. Alliez, P., Desbrun, M.: Progressive Compression for Lossless Transmission of Triangle Meshes. In: SIGGRAPH, pp. 195–202. ACM Press, New York (2001)
11. Choe, S., Kim, J., Lee, H., Lee, S.: Random Accessible Mesh Compression Using Mesh Chartification. IEEE Transactions on Visualization and Computer Graphics 15, 160–173 (2009)
12. Isenburg, M., Ivrissimtzis, I., Gumhold, S., Seidel, H.-P.: Geometry Prediction for High Degree Polygons. In: 21st Spring Conference on Computer Graphics (SCCG 2005), pp. 147–152. ACM Press, New York (2005)
13. Yoon, M., Ivrissimtzis, I., Lee, S.: Variational Bayesian Noise Estimation of Point Sets. Computers & Graphics 33, 226–234 (2009)
14. Ivrissimtzis, I.: Effects of Noise on Quantized Triangle Meshes. In: Dæhlen, M., Floater, M., Lyche, T., Merrien, J.-L., Mørken, K., Schumaker, L.L. (eds.) MMCS 2008. LNCS, vol. 5862, pp. 274–284. Springer, Heidelberg (2010)
15. Pham, H.: Springer Handbook of Engineering Statistics. Springer, London (2006)
16. Chao, M.-W., Lin, C.-H., Yu, C.-W., Lee, T.-Y.: A High Capacity 3D Steganography Algorithm. IEEE Transactions on Visualization and Computer Graphics 15, 274–284 (2009)
17. Yang, Y., Ivrissimtzis, I.: Polygonal Mesh Watermarking Using Laplacian Coordinates. Computer Graphics Forum (Eurographics/ACM SIGGRAPH Symposium on Geometry Processing 2010) 29, 1585–1593 (2010)

On Single Image Scale-Up Using Sparse-Representations

Roman Zeyde, Michael Elad, and Matan Protter

The Computer Science Department
Technion, Israel Institute of Technology, Haifa 32000, Israel
{romanz,elad,matanpr}@cs.technion.ac.il

Abstract. This paper deals with the single image scale-up problem using sparse-representation modeling. The goal is to recover an original image from its blurred and down-scaled noisy version. Since this problem is highly ill-posed, a prior is needed in order to regularize it. The literature offers various ways to address this problem, ranging from simple linear space-invariant interpolation schemes (e.g., bicubic interpolation), to spatially-adaptive and non-linear filters of various sorts. We embark from a recently-proposed successful algorithm by Yang et. al. [1,2], and similarly assume a local *Sparse-Land* model on image patches, serving as regularization. Several important modifications to the above-mentioned solution are introduced, and are shown to lead to improved results. These modifications include a major simplification of the overall process both in terms of the computational complexity and the algorithm architecture, using a different training approach for the dictionary-pair, and introducing the ability to operate without a training-set by boot-strapping the scale-up task from the given low-resolution image. We demonstrate the results on true images, showing both visual and PSNR improvements.

1 Introduction

Many applications require resolution enhancement of images acquired by low-resolution sensors (e.g. for high-resolution displays), while minimizing visual artifacts. The single image scale-up[1] problem can be formulated as follows: denote the original high-resolution image as $\mathbf{y}_h \in \mathbb{R}^{N_h}$, represented as a vector of length N_h pixels. In addition, denote the blur and decimation operators as $\mathbf{H} : \mathbb{R}^{N_h} \rightarrow \mathbb{R}^{N_h}$ and $\mathbf{S} : \mathbb{R}^{N_h} \rightarrow \mathbb{R}^{N_l}$ (where $N_l < N_h$) respectively. It is assumed hereafter that $\mathbf{H}$ applies a known low-pass filter to the image, and $\mathbf{S}$ performs a decimation by an integer factor s, by discarding rows/columns from the input image. $\mathbf{z}_l \in \mathbb{R}^{N_l}$ is defined to be the low-resolution noisy version of the original image as

$$\mathbf{z}_l = \mathbf{SHy}_h + \mathbf{v}, \tag{1}$$

[1] This problem is often referred to in the literature as *super-resolution*. The term *super-resolution* may be confusing, as it is also used in the context of fusing several low-resolution images into one high-resolution result.[3].

for an additive i.i.d. white Gaussian noise, denoted by $\mathbf{v} \sim \mathcal{N}\left(0, \sigma^2\mathbf{I}\right)$.

Given $\mathbf{z}_l$, the problem is to find $\hat{\mathbf{y}} \in \mathbb{R}^{N_h}$ such that $\hat{\mathbf{y}} \approx \mathbf{y}_h$. Due to the Gaussianity of $\mathbf{v}$, the maximum-likelihood estimation is obtained by the minimization of $\|\mathbf{SH}\hat{\mathbf{y}} - \mathbf{z}_l\|_2$. However, since $\mathbf{SH}$ is rectangular with more columns than rows, it cannot be inverted stably, and there are infinitely many solutions that lead to a zero value in the above-mentioned least-squares term. Existing single-image scale-up algorithms use various priors on the image in order to stabilize this inversion, ranging from Tikhonov regularization, robust statistics and Total-Variation, sparsity of transform coefficients, and all the way to example-based techniques that use training set of images as priors. While we do not provide a comprehensive review of these techniques, we refer the reader to the following papers [4,5,6,7,8,9,10,11].

In this work we shall use the *Sparse-Land* local model, as introduced in [12,13,14,15], for the scale-up problem. This model assumes that each patch from the images considered can be well represented using a linear combination of few atoms from a dictionary. Put differently, each patch is considered to be generated by multiplying a dictionary by a sparse (mostly zero) vector of coefficients. This assumption will help us in developing an algorithm for image scale-up. It is important to note that this is also the path taken by [1,2] and similar to [16,17]. However, our work differs from their solution in several important aspects, as described in the paper.

This paper is organized as follows: The incorporation of the *Sparse-Land* model into the scale-up problem is shown in Section 2. Section 3 describes the actual implementation details of the algorithm. Experiments and comparative results are given in Section 4, and conclusions are drawn in Section 5.

Just before we embark to the journey of handling the image scale-up problem in this work, we should refer to a delicate matter of the involvement of the blur that operates on the high-resolution image before the sub-sampling. In most cases, when an image is scaled-down, this process is necessarily accompanied by some pre-filter that averages local pixels, in order to reduce aliasing effects. Our work assumes that this is indeed the case, and the role of the scale-up process we aim to develop is to reverse both degradation steps – blur and sub-sampling. As such, our work considers a task that is a generalization of the deblurring problem, and by assuming no sub-sampling, the algorithm we derive here becomes yet-another deblurring method.[2]

There exists a different line of work on the image scale-up problem that strive to separate the treatment of the deblurring and the up-sampling problems. Such work would assume that there is no blur involved, so that the inversion process is purely an interpolation task. Alternatively, if there is a blur, the recovery performance would be measured with respect to the blurred high-resolution image. The rationale of such work is that once the image has been scaled-up in the best possible way, a deblurring stage should be used to get the final outcome. This

[2] In fact, the work reported in [17] does exactly that – developing a deblurring algorithm based on this paradigm.

is the approach taken, for example, in [18,19], and this explains why we cannot compare our results to theirs.

We should note that, since scaled-down images are typically blurred prior to sub-sampling, a separate treatment of the sampling and blur is necessarily sub-optimal, compared to a joint treatment, as done in [1,2] and our work. Indeed, after whatever (plain or smart) interpolation scheme, the additive noise in the resulting high-resolution image cannot be assumed as homogeneous, implying that off-the-shelf deblurring algorithms may perform poorly.

2 Incorporating the Sparse-Land Prior

In order to avoid the complexities caused by the different resolutions between $\mathbf{z}_l$ and $\mathbf{y}_h$, and in order to simplify the overall recovery algorithm, it is assumed hereafter that the image $\mathbf{z}_l$ is scaled-up by a simple interpolation operator $\mathbf{Q} : \mathbb{R}^{N_l} \rightarrow \mathbb{R}^{N_h}$ (e.g. bicubic interpolation) that fills-in the missing rows/columns, returning to the size of $\mathbf{y}_h$. This decision will not badly influence the computational complexity of the algorithm, and in fact, the eventual scale-up algorithm proposed here is much faster than the one proposed in [1,2,16]. The scaled-up image shall be denoted by $\mathbf{y}_l$ and it satisfies the relation

$$\mathbf{y}_l = \mathbf{Q}\mathbf{z}_l = \mathbf{Q}\left(\mathbf{S}\mathbf{H}\mathbf{y}_h + \mathbf{v}\right) = \mathbf{Q}\mathbf{S}\mathbf{H}\mathbf{y}_h + \mathbf{Q}\mathbf{v} = \mathbf{L}^{all}\mathbf{y}_h + \tilde{\mathbf{v}}. \tag{2}$$

The goal is to process $\mathbf{y}_l \in \mathbb{R}^{N_h}$ and produce a result $\hat{\mathbf{y}}_h \in \mathbb{R}^{N_h}$, which will get as close as possible to the original high-resolution image, $\mathbf{y}_h \in \mathbb{R}^{N_h}$.

The algorithm we propose operates on patches extracted from $\mathbf{y}_l$, aiming to estimate the corresponding patch from $\mathbf{y}_h$. Let $\mathbf{p}_h^k = \mathbf{R}_k\mathbf{y}_h \in \mathbb{R}^n$ be a high-resolution image patch of size $\sqrt{n} \times \sqrt{n}$, extracted by the operator $\mathbf{R}_k : \mathbb{R}^{N_h} \rightarrow \mathbb{R}^n$ from the image $\mathbf{y}_h$ in location k. It is assumed that the locations to consider $\{k\}$ are only those centered around true pixels in the low-resolution image $\mathbf{y}_l$ (as opposed to filled-in pixels due to the interpolation). This set of samples is referred to hereafter as the set Ω.

It is now time to invoke the *Sparse-Land* model: it shall be further assumed that $\mathbf{p}_h^k \in \mathbb{R}^n$ can be represented sparsely by $\mathbf{q}^k \in \mathbb{R}^m$ over the dictionary $\mathbf{A}_h \in \mathbb{R}^{n\times m}$, namely:

$$\mathbf{p}_h^k = \mathbf{A}_h\mathbf{q}^k, \tag{3}$$

where $\|\mathbf{q}^k\|_0 \ll n$, where the ℓ_0-pseudo-norm counts the number of non-zeros in the vector $\mathbf{q}^k$. The matrix A_h is the dictionary that characterizes the high-resolution patches. Its construction will be discussed in details in Section 3.

Consider the corresponding low-resolution patch $\mathbf{p}_l^k = \mathbf{R}_k\mathbf{y}_l$, extracted from $\mathbf{y}_l$ in the same location (the patches $\mathbf{p}_l^k$ and $\mathbf{p}_h^k$ are centered around the same pixel k), such that its size is $\sqrt{n}\times\sqrt{n}$. Since the operator $\mathbf{L}^{all} = \mathbf{QSH}$ transforms the complete high-resolution image $\mathbf{y}_h$ to the low-resolution one, $\mathbf{y}_l$, it can be assumed that $\mathbf{p}_l^k = \mathbf{L}\mathbf{p}_h^k + \tilde{\mathbf{v}}_k$, where $\mathbf{L}$ is a local operator being a portion of $\mathbf{L}^{all}$, and $\tilde{\mathbf{v}}_k$ is the additive noise in this patch. Note that $\mathbf{L}$ is a spatially independent operator, as only locations $k \in \Omega$ from $\mathbf{y}_l$ are considered.[3]

Since it is assumed that $\mathbf{p}_h^k = \mathbf{A}_h \mathbf{q}^k$, multiplication of this equation by $\mathbf{L}$ gives

$$\mathbf{L}\mathbf{p}_h^k = \mathbf{L}\mathbf{A}_h \mathbf{q}^k. \tag{4}$$

Exploiting the relation between the low-resolution and the high-resolution patches $\mathbf{p}_l^k = \mathbf{L}\mathbf{p}_h^k + \tilde{\mathbf{v}}_k$, we obtain

$$\mathbf{L}\mathbf{A}_h \mathbf{q}^k = \mathbf{L}\mathbf{p}_h^k = \mathbf{p}_l^k - \tilde{\mathbf{v}}_k, \tag{5}$$

implying that

$$\|\mathbf{p}_l^k - \mathbf{L}\mathbf{A}_h \mathbf{q}^k\|_2 \leq \epsilon, \tag{6}$$

where ϵ is related to the noise power σ of $\mathbf{v}$.

The key observation from the above derivations is that the low-resolution patch $\mathbf{p}_l^k$ can be represented by the same sparse vector $\mathbf{q}^k$ over the effective dictionary $\mathbf{A}_l = \mathbf{L}\mathbf{A}_h$, with a controlled error ϵ. This implies that for a given low-resolution patch $\mathbf{p}_l^k$, its sparse representation vector, $\mathbf{q}^k$, is found and then $\mathbf{p}_h^k$ can be recovered by simply multiplying this representation by the dictionary $\mathbf{A}_h$. This is the core idea behind the image scale-up algorithm as developed by Yang et. al, [1,2], and we follow it as well, with important modifications.

We should comment that the above observation is fragile and may become wrong for some image patches. Even if the dictionary $\mathbf{A}_h$ has low-coherence, the multiplication by $\mathbf{L}$ is expected to cause a deterioration (i.e. increase) in the mutual coherence of $\mathbf{A}_l = \mathbf{L}\mathbf{A}_h$. This may lead to the detection of a wrong sparse representation vector for a low-resolution patch, which does not fit the high-resolution one [14].

Similar to the approach taken in [1,2], we disregard this problem, with the hope that such errors are rarely encountered. To our aid come two possible forces: (i) Even if there is a mistake in the sparse coding of a low-resolution patch, it is unclear whether such a error reflects an error in the *recovered* high-resolution patch (even though we rely on the wrong sparse representation vector) – this matter calls for deeper theoretical study, which is beyond the scope of this work. Furthermore, (ii) As we explain in the next section, we work with overlapping patches that are averaged, and therefore visual artifacts in a high-resolution recovered patch may be attenuated by nearby patches that get better treatment. Naturally, the empirical results we show towards the end of this paper should serve as the ultimate judge, whether our assumptions are justified.

[3] The observant reader might be troubled by boundary issues because of the spatial extent of the operator $\mathbf{L}_{all}$, and the fact that we have chosen the low-resolution and the high-resolution patches to be of the same size. However, the developed algorithm will not make use of the operator $\mathbf{L}$, and bypass this issue - more on this matter is brought in the next section.

3 The Proposed Single-Image Scale-Up Algorithm

The scale-up algorithm consists of a training phase (that can be done off-line) as described in Figure 1 and a reconstruction phase, performing the scale-up on the test image using the trained model from the previous phase, as described in Figure 2.

1. Training set construction: A set of high-resolution training images $\{\mathbf{y}_h^j\}_j$ is collected, Low-resolution images $\{\mathbf{y}_l^j\}_j$ are constructed using scale-down operator $\mathbf{L}_{all}$ and pairs of matching patches that form the training database, $\mathcal{P} = \{\mathbf{p}_h^k, \mathbf{p}_l^k\}_k$, are extracted.
2. Each of these patch-pairs undergoes a pre-processing stage that removes the low-frequencies from $\mathbf{p}_h^k$ and extracts features from $\mathbf{p}_l^k$.
3. Dimensionality reduction is applied on the features of the low-resolution patches $\mathbf{p}_l^k$, making the dictionary training step much faster.
4. A dictionary $\mathbf{A}_l$ is trained for the low-resolution patches, such that they can be represented sparsely.
5. A corresponding dictionary $\mathbf{A}_h$ is constructed for the high-resolution patches, such that it matches the low-resolution one.

Fig. 1. Proposed algorithm's summary: training phase

1. Given a test low-resolution image $\mathbf{z}_l$ to be scaled-up, it is interpolated to $\mathbf{y}_l$ of the destination size, and all that it requires is a spatial non-linear filtering to sharpen it.
2. Pre-processed patches $\mathbf{p}_l^k$ are extracted from each location $k \in \Omega$, and then sparse-coded using the trained dictionary $\mathbf{A}_l$.
3. The found representations $\{\mathbf{q}^k\}$ are then used to recover the high-resolution patches by multiplying them with $\mathbf{A}_h$.
4. The recovered high-resolution patches $\{\mathbf{p}_h^k\}$ are finally merged by averaging in the overlap area to create the resulting image.

Fig. 2. Proposed algorithm's summary: reconstruction phase

3.1 Training Set Construction

The training phase starts by collecting several images $\{\mathbf{y}_h^j\}_j$, which are considered to be the high-resolution examples. Each of these images is blurred and down-scaled by a factor of s. This leads to the formation of the corresponding low-resolution images $\{\mathbf{z}_l^j\}_j$, which are then scaled up back to the original size using $\mathbf{Q}$, resulting with the set $\{\mathbf{y}_l^j\}_j$. Thus,

$$\mathbf{y}_l^j = \mathbf{L}_{all}\mathbf{y}_h^j + \tilde{\mathbf{v}}^j. \tag{7}$$

It is important to note that the same operators $\mathbf{S}$, $\mathbf{H}$ and $\mathbf{Q}$ should be used both in the training and the reconstruction phases.

3.2 Preprocessing and Feature Extraction

The next step is pre-processing by high-pass filtering, similar to the approach in [7,1,2,16]. Rather than extracting small image-patches and applying this step on them, the desired pre-processing is employed directly on the full images, and only then are the patches extracted. This order avoids boundary problems due to the small patch size.[4]

The pre-processing applied to the high-resolution images consists of a removal of their low-frequencies, which is done by computing the difference images $\mathbf{e}_h^j = \mathbf{y}_h^j - \mathbf{y}_l^j$. The reason for this step is the desire to focus the training on characterizing the relation between the low-resolution patches and the edges and texture content within the corresponding high-resolution ones.

As for the pre-processing of the low-resolution images, these are filtered using R high-pass filters, in order to extract local features that correspond to their high-frequency content. Thus, each low-resolution image $\mathbf{y}_l^j$ results in a set of R filtered images, $\{f_r * \mathbf{y}_l^j\}_r$ for $r = 1, 2, \ \ldots \ , R$ (where $*$ stands for a convolution). Typical filters to be used may be gradient and Laplacian high-pass filters.

After the two pre-processing steps described above, local patches are extracted forming the data-set $\mathcal{P} = \{\mathbf{p}_h^k, \mathbf{p}_l^k\}_k$. Considering only locations $k \in \Omega$, $\mathbf{p}_h^k$ patches of size $\sqrt{n} \times \sqrt{n}$ pixels are extracted from the high-resolution images $\mathbf{e}_h^j$. The corresponding low-resolution $\mathbf{p}_l^k$ patches are extracted from the same locations in the filtered images $f_k * \mathbf{y}_l^j$ and using the same size ($\sqrt{n} \times \sqrt{n}$ pixels). Thus, every corresponding R such low-resolution patches are concatenated into one vector $\tilde{\mathbf{p}}_l^k$ of length nR. Note that the high-resolution patch size should be at least of size $s \times s$ so as to cover the high-resolution image. A larger patch-size results in overlaps between patches, which improves the reconstruction result (by reducing errors and discontinuities between reconstructed patches).

3.3 Dimensionality Reduction

The formed low-resolution patches, started as $\sqrt{n}/s \times \sqrt{n}/s = n/s^2$ pixel patches (in the images $\mathbf{z}_l^j$), are now represented as $\tilde{\mathbf{p}}_l^k$ of nR dimensions after an interpolation operator $\mathbf{Q}$ and set of R linear filters. As a result, the intrinsic dimensionality (n/s^2) of the resulting patches should not increase and it is much smaller than the representation dimension (nR), resulting in superfluous computations. The advantage of performing a dimensionality reduction is saving computations in the subsequent training and super-resolution algorithms. Therefore, the last step before turning to the dictionary learning stage is reducing the dimension

[4] A patch of size $\sqrt{n} \times \sqrt{n}$ in $\mathbf{y}_l$ should correspond to a larger patch in $\mathbf{y}_h$, because of the spatial extent of the blur and the scale-up operations. Nevertheless, this additional "band" of pixels can be disregarded, concentrating on predicting only the center portions of the destination patch from $\mathbf{y}_h$.

of the input low-resolution patches, $\{\tilde{\mathbf{p}}_l^k\}_k$. The Principal Component Analysis (PCA) algorithm is applied on these vectors, seeking a subspace on which these patches could be projected while preserving 99.9% of their average energy. The projection operator that transforms the patch $\tilde{\mathbf{p}}_l^k \in \mathbb{R}^{nR}$ to its reduced feature vector, $\mathbf{p}_l^k \in \mathbb{R}^{n_l}$, is denoted by $\mathbf{B} \in \mathbb{R}^{n_l \times nR}$, $\mathbf{p}_l^k = \mathbf{B}\tilde{\mathbf{p}}_l^k$.

3.4 Dictionary Learning

The starting point of the dictionary learning stage are the low-resolution patches $\{\mathbf{p}_l^k\}_k \subseteq \mathbb{R}^{n_l}$. The K-SVD dictionary training procedure [20] is applied to these patches, resulting in the dictionary $\mathbf{A}_l \in \mathbb{R}^{n_l \times m}$:

$$\mathbf{A}_l, \{\mathbf{q}^k\} = \underset{\mathbf{A}_l, \{\mathbf{q}^k\}}{\operatorname{argmin}} \sum_k \|\mathbf{p}_l^k - \mathbf{A}_l \mathbf{q}^k\|^2 \text{ s.t. } \|\mathbf{q}^k\|_0 \le L \quad \forall k. \tag{8}$$

A side product of this training is the sparse representation coefficients vectors $\{\mathbf{q}^k\}_k$ that correspond to the training patches $\{\mathbf{p}_l^k\}_k$.

The next step is the high-resolution dictionary construction. Recall that our intention is to recover the patch $\mathbf{p}_h^k$ by approximating it as being $\mathbf{p}_h^k \approx \mathbf{A}_h \mathbf{q}^k$. Effectively, the found sparse representation vector for the low-resolution patch is multiplied by the high-resolution dictionary for recovering $\mathbf{p}_l^k$. The dictionary $\mathbf{A}_h$ is therefore sought such that this approximation is as exact as possible. Thus, this dictionary is defined to be the one that minimizes the mean approximation error, i.e.,

$$\begin{aligned} \mathbf{A}_h &= \underset{\mathbf{A}_h}{\operatorname{argmin}} \sum_k \|\mathbf{p}_h^k - \mathbf{A}_h \mathbf{q}^k\|_2^2 \\ &= \underset{\mathbf{A}_h}{\operatorname{argmin}} \|\mathbf{P}_h - \mathbf{A}_h \mathbf{Q}\|_F^2, \end{aligned} \tag{9}$$

where the matrix $\mathbf{P}_h$ is constructed with the high-resolution training patches $\{\mathbf{p}_h^k\}_k$ as its columns, and similarly, $\mathbf{Q}$ contains $\{\mathbf{q}^k\}_k$ as its columns. We note that this is also the approach taken in [16]. The solution of the problem is given by the following Pseudo-Inverse expression (given that $\mathbf{Q}$ has full row rank):

$$\mathbf{A}_h = \mathbf{P}_h \mathbf{Q}^+ = \mathbf{P}_h \mathbf{Q}^T (\mathbf{Q}\mathbf{Q}^T)^{-1}. \tag{10}$$

Note that the above approach disregards the fact that the high-resolution patches overlap. Thus, a better (and more complex) training procedure can be envisioned for computing $\mathbf{A}_h$. Since the eventual high-resolution image (in the reconstruction stage) is constructed by positioning high-resolution patches and averaging them, $\mathbf{A}_h$ should be optimized such that the resulting image is as close as possible to the original one.

Define the operator $\mathbf{R}_k$, which extracts a patch of size $n \times n$ from the high resolution image in location k. The image $\hat{\mathbf{y}}_h$ should be constructed by the following formula [12,13]:

$$\hat{\mathbf{y}}_h = \mathbf{y}_l + \left[\sum_{k \in \Omega} \mathbf{R}_k^T \mathbf{R}_k\right]^{-1} \left[\sum_{k \in \Omega} \mathbf{R}_k^T \mathbf{A}_h \mathbf{q}^k\right]. \tag{11}$$

The term $\mathbf{R}_k^T \mathbf{A}_h \mathbf{q}^k$ builds the high-resolution patch, $\mathbf{A}_h \mathbf{q}^k$, and then positions it in the k-th location in the high-resolution image. The term $\mathbf{W} = \sum_k \mathbf{R}_k^T \mathbf{R}_k \in \mathbb{R}^{N_h \times N_h}$ is a diagonal matrix that weights every pixel in the high-resolution outcome, based on the number of contributions it gets form the overlapped patches. Note that $\mathbf{y}_l$ appears in the error computation, due to the fact that the patches in $\mathbf{P}_h$ are constructed from the difference images $\mathbf{e}_h = \mathbf{y}_h - \mathbf{y}_l$, and this means that for the image $\hat{\mathbf{y}}_h$ to be constructed, the algorithm should return these low-frequencies.

Based on the above, it is natural to define the best dictionary $\mathbf{A}_h$ as the solution of the optimization task:

$$\begin{aligned} \mathbf{A}_h &= \arg\min_{\mathbf{A}_h} \|\mathbf{y}_h - \hat{\mathbf{y}}_h\|_2^2 \qquad (12) \\ &= \arg\min_{\mathbf{A}_h} \left\| \mathbf{y}_h - \mathbf{y}_l - \left[\sum_{k\in\Omega} \mathbf{R}_k^T \mathbf{R}_k \right]^{-1} \left[\sum_{k\in\Omega} \mathbf{R}_k^T \mathbf{A}_h \mathbf{q}^k \right] \right\|_2^2 . \end{aligned}$$

Denote $\mathbf{W}_k = \mathbf{R}_k \mathbf{W}^{-1} \in \mathbb{R}^{n \times N_h}$ and write $\hat{\mathbf{y}}_h = \mathbf{y}_l + \sum_k \mathbf{W}_k^T \mathbf{A}_h \mathbf{q}^k$. The goal is to minimize $\|\mathbf{y}_h - \hat{\mathbf{y}}_h\|_2^2$ with respect to $\mathbf{A}_h$. Denote $\mathbf{e}_h = \mathbf{y}_h - \mathbf{y}_l$, and $\mathbf{A}_h$ is obtained by the minimization of

$$\mathbf{A}_h = \arg\min_{\mathbf{A}_h} \|\mathbf{y}_h - \mathbf{y}_l - \sum_k \mathbf{W}_k^T \mathbf{A}_h \mathbf{q}^k\|_2^2 = \arg\min_{\mathbf{A}_h} \|\mathbf{e}_h - \sum_k \mathbf{W}_k^T \mathbf{A}_h \mathbf{q}^k\|_2^2 \qquad (13)$$

Given $\mathbf{X} \in \mathbb{R}^{n \times m}$, define $\mathbf{x} \equiv \mathbf{cs}(\mathbf{X})$ to be the column-stack version of $\mathbf{X}$ (using $\mathbf{x}_{i+nj} = \mathbf{X}_{ij}$). Now using the Kronecker product property: $\mathbf{cs}(\mathbf{BAC}) = \left(\mathbf{C}^T \otimes \mathbf{B}\right) \mathbf{cs}(\mathbf{A}) = \left(\mathbf{C} \otimes \mathbf{B}^T\right)^T \mathbf{cs}(\mathbf{A})$, we obtain

$$\mathbf{cs}(\mathbf{e}_h) = \mathbf{e}_h = \sum_k \mathbf{W}_k^T \mathbf{A}_h \mathbf{q}^k = \left(\sum_k \mathbf{q}^k \otimes \mathbf{W}_k \right)^T \mathbf{cs}(\mathbf{A}_h) = \mathbf{M} \cdot \mathbf{cs}(\mathbf{A}_h), \qquad (14)$$

where $\mathbf{M} \in \mathbb{R}^{N_h \times mn}$ is defined as $\mathbf{M}^T = \sum_k \mathbf{q}^k \otimes \mathbf{W}_k$. Therefore, one way to get the optimal $\mathbf{A}_h$ is by the direct formula, $\mathbf{M}^\dagger \mathbf{e}_h$.

Since the matrices involved may be too large, we present an alternative, iterative, approach. Note that the gradient of $f(\mathbf{A}_h) = \frac{1}{2}\|\mathbf{e}_h - \sum_k \mathbf{W}_k^T \mathbf{A}_h \mathbf{q}^k\|_2^2$ with respect to $\mathbf{A}_h$, can be written as

$$\nabla_{\mathbf{A}_h} f = \sum_k \mathbf{W}_k \left(\mathbf{e}_h - \sum_k \mathbf{W}_k^T \mathbf{A}_h \mathbf{q}^k \right) \left(\mathbf{q}^k\right)^T . \qquad (15)$$

An iterative scheme (such as the Conjugate Gradient method) can be used to find the optimal $\mathbf{A}_h$, using the gradient expression above.

The dictionary resulting from the training process is expected to better reconstruct the output result. In the experiments given below, both ways to derive $\mathbf{A}_h$ are adopted.

The two corresponding dictionaries $\{\mathbf{A}_l, \mathbf{A}_h\}$ conclude the training phase of the super-resolution algorithm, that started with the high-resolution training set $\{\mathbf{y}_h^j\}_j$.

3.5 Reconstruction Phase

The reconstruction phase attempts to magnify a low-resolution image $\mathbf{z}_l$. This image is assumed to have been generated from a high-resolution image $\mathbf{y}_h$ by the same blur and scale-down operations as used in the training. The following steps are performed:

1. The image is scaled up by a factor of s using bicubic interpolation $\mathbf{Q}$, resulting with $\mathbf{y}_l \in \mathbb{R}^{n_l}$.
2. The image $\mathbf{y}_l$ is filtered using the same R high-pass filters that were used for feature extraction in the training, obtaining $f_k * \mathbf{y}_l$.
3. Patches are extracted from these R images, each of size $\sqrt{n} \times \sqrt{n}$ from locations $k \in \Omega$. Every R such patches that correspond to the same location are concatenated to form a patch vector $\tilde{\mathbf{p}}_l^k$. This collection of patches forms the set $\{\tilde{\mathbf{p}}_l^k\}_k$.
4. The patches $\{\tilde{\mathbf{p}}_l^k\}_k$ are multiplied by the projection operator $\mathbf{B}$ for dimensionality reduction, resulting with the set $\{\mathbf{p}_l^k\}_k$, each patch of length n_l (for $n = 81$ and a scale factor $s = 3$, our tests lead to $n_l \approx 30$).
5. The OMP algorithm is applied to $\{\mathbf{p}_l^k\}_k$, allocating L atoms to their representation, and finding the sparse representation vectors $\{\mathbf{q}^k\}_k$.
6. The representation vectors $\{\mathbf{q}^k\}_k$ are multiplied by the high-resolution dictionary $\mathbf{A}_h$, and the approximated high-resolution patches, $\{\mathbf{A}_h\mathbf{q}^k\}_k = \{\hat{\mathbf{p}}_h^k\}_k$ are obtained.
7. The final super-resolved image $\hat{\mathbf{y}}_h$ is constructed from $\hat{\mathbf{p}}_h^k$ by solving the following minimization problem with respect to $\hat{\mathbf{y}}_h$:

$$\hat{\mathbf{y}}_h = \underset{\hat{\mathbf{y}}_h}{\operatorname{argmin}} \sum_k \left\| \mathbf{R}_k(\hat{\mathbf{y}}_h - \mathbf{y}_l) - \hat{\mathbf{p}}_h^k \right\|_2^2 . \tag{16}$$

   This problem states that extracted patches from the resulting difference image, $\hat{\mathbf{y}}_h - \mathbf{y}_l$, should be as close as possible to the approximated patches, $\hat{\mathbf{p}}_h^k$. This problem has a closed-form Least-Squares solution, given by

$$\hat{\mathbf{y}}_h = \mathbf{y}_l + \left[\sum_k \mathbf{R}_k^T \mathbf{R}_k\right]^{-1} \sum_k \mathbf{R}_k^T \hat{\mathbf{p}}_h^k, \tag{17}$$

   which was already mentioned above. This seemingly complex term is actually very simple – it is equivalent to putting $\hat{\mathbf{p}}_h^k$ in their proper locations, averaging in overlap regions, and adding $\mathbf{y}_l$ to get the final image $\hat{\mathbf{y}}_h$.

3.6 Bootstrapping Approach

If the training process has no access to an external set of images, the algorithm may be adapted to train and "bootstrap" itself from a single test image, as proposed by [22]. Note that in order to train the dictionaries $\{\mathbf{A}_l, \mathbf{A}_h\}$, the proposed algorithm needs only access to pairs of low-resolution and high-resolution images. Using the test image $\mathbf{z}_l$ as the "high-resolution" image and its scaled-down version $\mathbf{z}_{ll}$ (by an appropriate choice of $\mathbf{S}$ and $\mathbf{H}$), the algorithm can be easily extended to perform "bootstrapping" from a single image:

– The training phase is applied to $\mathbf{z}_l$ (as high-resolution image), $\mathbf{z}_{ll}$ (as low-resolution image).
– The trained dictionaries are used to enable reconstruction phase, which scales up $\mathbf{z}_l$ into $\mathbf{y}_h$.

Thus, it is possible to scale-up a single image by learning the *Sparse-Land* model directly from the test image itself, provided that the training process has enough training data to build a valid *Sparse-Land* model from.

4 Results

In this section we provide series of tests of the proposed algorithm, and comparisons to the results obtained in [1,2]. In all experiments, the dictionary $\mathbf{A}_h$ is trained using the simpler method (that does not take the overlaps into account), unless mentioned differently.

4.1 Text Scale-Up

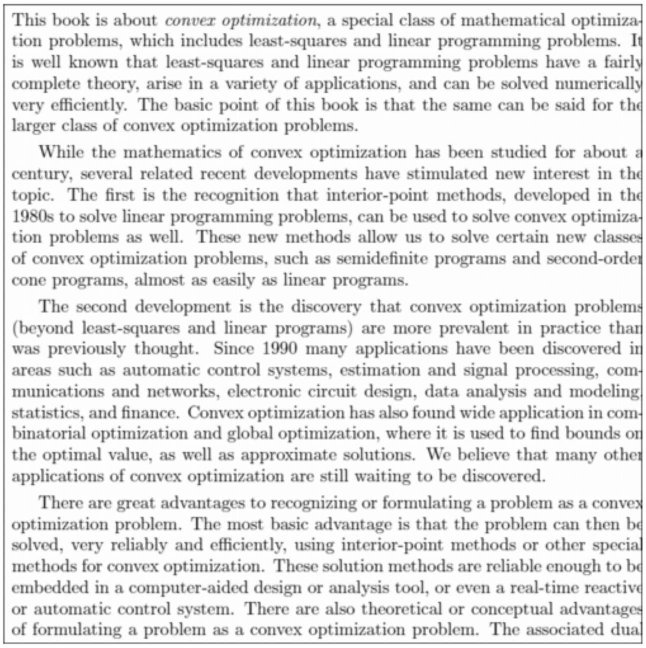
The first test contains images showing a printed text. The training image (screen-grabbed) is shown in Figure 3. Only this image is used for the training, and we expect that adding more images would lead to improved results. The global operator $\mathbf{L}_{all}$ in this experiment is implemented by first blurring the high-resolution images $\mathbf{y}_h^j$ with a 1D filter $[1,3,4,3,1]/12$ both horizontally and vertically, and then down-sampling it by a factor of $s=3$, i.e., the scaled-down image $\mathbf{z}_l$ is one-ninth of the original image size. The image $\mathbf{y}_l$ is created by bicubic interpolation of $\mathbf{z}_l$, returning to the original size.

Extraction of features from the low-resolution images is done exactly as proposed in [1,2] using 4 filters that perform 1-st and 2-nd horizontal and vertical derivatives: $f_1 = [\mathtt{1}, -\mathtt{1}] = f_2^T$ and $f_3 = [\mathtt{1}, -\mathtt{2}, \mathtt{1}] = f_4^T$. These filters are applied such that only sampled pixels are used in the filtering computation[5]. The patch size used is $n = 81$, and the PCA results with a reduction from $4 \cdot 81 = 324$ dimensions to $n_l = 30$ dimensions. The dictionary training procedure applied 40 iterations of the K-SVD algorithm, with $m = 1{,}000$ atoms in the dictionary, and allocating $L = 3$ atoms for each representation vector.

The test image (a different image, grabbed from a different page, but having the same scale) is shown in Figure 4. This figure shows the original test image, and the scaled-down version that should be scaled-up. The scaling-up results are also shown in Figure 4, and it is evident that the outcome is far better, compared to the bicubic interpolation[6], showing Peak-SNR (PSNR) improvement of

[5] This means that either $\mathbf{z}_l$ is filtered and then interpolated, or $\mathbf{y}_l$ is filtered with zero-padded filters of the form $f_1 = [\mathtt{0},\mathtt{0},\mathtt{1},\mathtt{0},\mathtt{0},-\mathtt{1}] = f_2^T$ and $f_3 = [\mathtt{1},\mathtt{0},\mathtt{0},-\mathtt{2},\mathtt{0},\mathtt{0},\mathtt{1}] = f_4^T$.

[6] Since bicubic interpolation does not include a deblurring capability, it seems that comparing our results to it is unfair. However, when trying to deblur the bicubic interpolation result (using TV-deblurring), we found out that the quality deteriorates. This could be explained by the non-homogeneous nature of the noise that is magnified in an uncontrolled way by the deblurring.


This book is about *convex optimization*, a special class of mathematical optimization problems, which includes least-squares and linear programming problems. It is well known that least-squares and linear programming problems have a fairly complete theory, arise in a variety of applications, and can be solved numerically very efficiently. The basic point of this book is that the same can be said for the larger class of convex optimization problems.

While the mathematics of convex optimization has been studied for about a century, several related recent developments have stimulated new interest in the topic. The first is the recognition that interior-point methods, developed in the 1980s to solve linear programming problems, can be used to solve convex optimization problems as well. These new methods allow us to solve certain new classes of convex optimization problems, such as semidefinite programs and second-order cone programs, almost as easily as linear programs.

The second development is the discovery that convex optimization problems (beyond least-squares and linear programs) are more prevalent in practice than was previously thought. Since 1990 many applications have been discovered in areas such as automatic control systems, estimation and signal processing, communications and networks, electronic circuit design, data analysis and modeling, statistics, and finance. Convex optimization has also found wide application in combinatorial optimization and global optimization, where it is used to find bounds on the optimal value, as well as approximate solutions. We believe that many other applications of convex optimization are still waiting to be discovered.

There are great advantages to recognizing or formulating a problem as a convex optimization problem. The most basic advantage is that the problem can then be solved, very reliably and efficiently, using interior-point methods or other special methods for convex optimization. These solution methods are reliable enough to be embedded in a computer-aided design or analysis tool, or even a real-time reactive or automatic control system. There are also theoretical or conceptual advantages of formulating a problem as a convex optimization problem. The associated dual

Fig. 3. Text experiment: The training image for the image scaling-up algorithm. This image is of size 717×717 pixels, and it provides a set of $54,289$ training patch-pairs.

2.27dB, with PSNR computed by

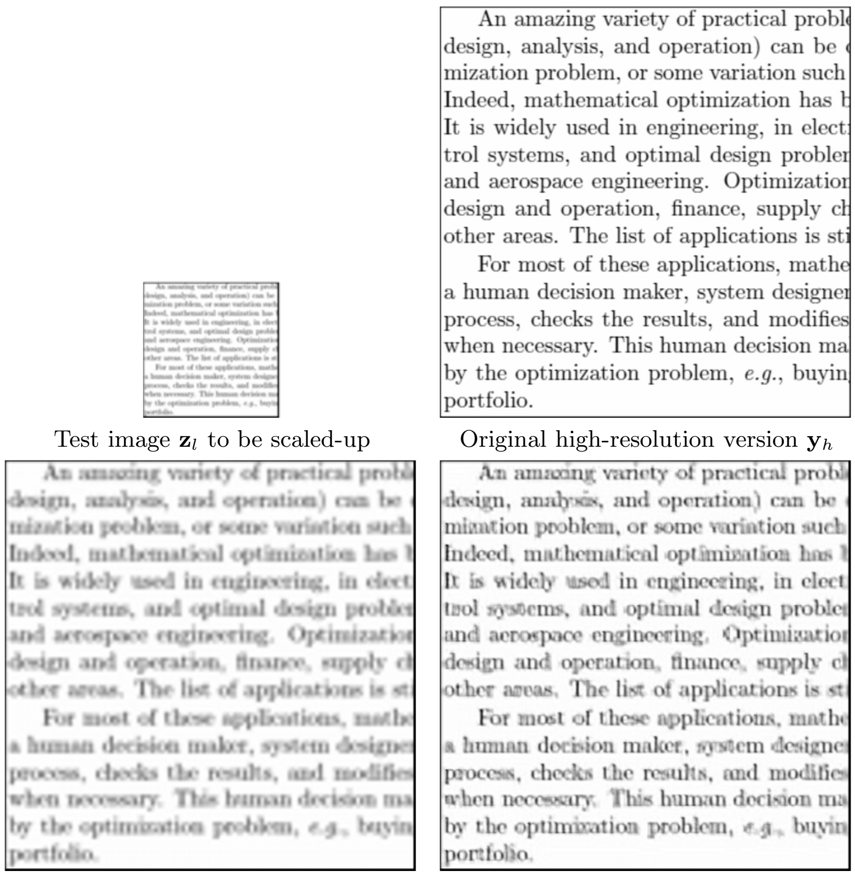
$$\text{PSNR} = 10 \log_{10} \left(\frac{255^2 \cdot N}{\sum_i (\hat{\mathbf{y}}_i - \mathbf{y}_i)^2} \right), \tag{18}$$

with $\mathbf{y}, \hat{\mathbf{y}} \in [0, 255]^N \subseteq \mathbb{R}^N$.

4.2 PSNR Comparison with Yang et. al. [1,2]

The second experiment aims to give a comprehensive comparison between the results of our algorithm and the one in [1,2]. The proposed algorithm is implemented in MATLAB using optimized implementation for K-SVD and OMP algorithms [21], on Intel Core 2 Duo P8600 at 2.4GHz with 4GB of RAM. It is trained on the same training set used in [1,2], using $s = 3$ scale-up configuration. Each training image is blurred using a bicubic filter and decimated by a factor of s; feature extraction is done as before (using gradient and laplacian filters).

Around 130,000 training patch-pairs are collected and PCA is applied to reduce feature dimensions to $n_l = 30$. Low-resolution dictionary learning takes approximately 12 minutes for 40 iterations of the K-SVD algorithm, with $m = 1,000$ atoms in the dictionary, and allocating $L = 3$ atoms per patch-representation. Moreover, the high-resolution dictionary training takes just a few seconds,


Test image $\mathbf{z}_l$ to be scaled-up
Original high-resolution version $\mathbf{y}_h$
Bicubic interpolation $\mathbf{y}_l(RMSE{=}47.06)$
Proposed algorithm $\hat{\mathbf{y}}_h(RMSE{=}36.22)$

An amazing variety of practical probl
design, analysis, and operation) can be
mization problem, or some variation such
Indeed, mathematical optimization has b
It is widely used in engineering, in elect
trol systems, and optimal design probler
and aerospace engineering. Optimizatior
design and operation, finance, supply ch
other areas. The list of applications is sti

For most of these applications, mathe
a human decision maker, system designer
process, checks the results, and modifies
when necessary. This human decision ma
by the optimization problem, *e.g.*, buyin
portfolio.

Fig. 4. Text experiment: The image $\mathbf{z}_l$ is of size 120×120 pixels, and it provides a set of $12,996$ patches to operate on. RMSE is computed using $\frac{1}{N}\sqrt{\sum_{i=1}^{N} |\mathbf{y}_i - \hat{\mathbf{y}}_i|^2}$ where $\mathbf{y}, \hat{\mathbf{y}} \in [0, 255]^N$.

using the pseudo-inverse expression $\mathbf{A}_h = \mathbf{P}_h\mathbf{Q}^+$. The proposed training algorithm is much faster than the one used by Yang et. al. [1,2] (taking several hours to run on the same settings). The reconstruction algorithm is tested on 14 test images (taking a few seconds on each image, using fully overlapping 3×3 patches in low-resolution scale) and its results are compared versus bicubic interpolation and the reconstruction algorithm proposed by Yang et. al. [1,2]. The resulting images' boundary is cropped (to ignore boundary effects of overlap-and-add method) and Peak-SNR is computed.

The results are summarized in Table 1. A few 100×100 representative windows from different images are compared at Figure 5. On the left is the original image, followed by bicubic interpolation, Yang et. al. [1,2] and the proposed algorithm's results on the right, at the last column.

Table 1. PSNR comparison results. (i) corresponds to the simpler high-resolution dictionary training method, using Pseudo-Inverse expression (see Equation [10]). (ii) corresponds to the more complex method that takes care for overlapping patches.

Name	Bicubic	Yang et. al [1,2]	Proposed (i)	Proposed (ii)
`baboon`	23.2	23.5	23.5	23.5
`barbara`	26.2	26.4	26.8	26.7
`bridge`	24.4	24.8	25.0	25.0
`coastguard`	26.6	27.0	27.1	27.2
`comic`	23.1	23.9	24.0	24.1
`face`	32.8	33.1	33.5	33.6
`flowers`	27.2	28.2	28.4	28.6
`foreman`	31.2	32.0	33.2	33.6
`lenna`	31.7	32.6	33.0	33.1
`man`	27.0	27.8	27.9	28.0
`monarch`	29.4	30.7	31.1	31.4
`pepper`	32.4	33.3	34.1	34.2
`ppt3`	23.7	25.0	25.2	25.6
`zebra`	26.6	28.0	28.5	28.6
Average	27.5	28.3	28.7	28.8

The proposed algorithm performs visually much better than bicubic interpolation, and on some images considerably better than Yang et. al. [1,2] algorithm, having less visual artifacts and producing sharper results with improved PSNR. Moreover, the implementation of the proposed algorithm is much faster (by an order of magnitude) than Yang et. al. [1,2] implementation, using optimized K-SVD and OMP implementations by [21].

4.3 Bootstrapping Approach for Single Image Scale-Up

The third experiment is the image `Building`. Starting from the original image $\mathbf{y}_h$ of size 800×800 pixels, the image is filtered with the separable filter $[1, 2, 1]/4$ (horizontally and vertically), and down-scaled by a factor of $s = 2$ to obtain $\mathbf{z}_l$ of size 400×400.

In this experiment, the dictionaries are trained using the very same image, by further scaling it down by a factor $s = 2$, resulting with the image $\mathbf{z}_{ll}$ of size 200×200. The image pair $\{\mathbf{z}_l,\ \mathbf{z}_{ll}\}$ is used for the training, based on the expectation that the relation between these two images reflects also the relation that should be used to up-scale from $\mathbf{z}_l$ to $\mathbf{y}_h$.

Extraction of features from the low-resolution images is done using the same 4 filters, and the dimensionality reduction leads this time from $n = 81$ to $n_l = 42$. The training data contains $37,636$ pairs of low- and high-resolution patches to be modeled. The parameters of the dictionary training all remain the same (40 iterations of the K-SVD algorithm, $m = 1000$ atoms in the dictionary, and $L = 3$ atoms per representation).

Figure [6] shows the original image $\mathbf{y}_h$, the bicubic scaled-up image $\mathbf{y}_l$, and the result of the scaling up algorithm, $\hat{\mathbf{y}}_h$. The difference between the two images is

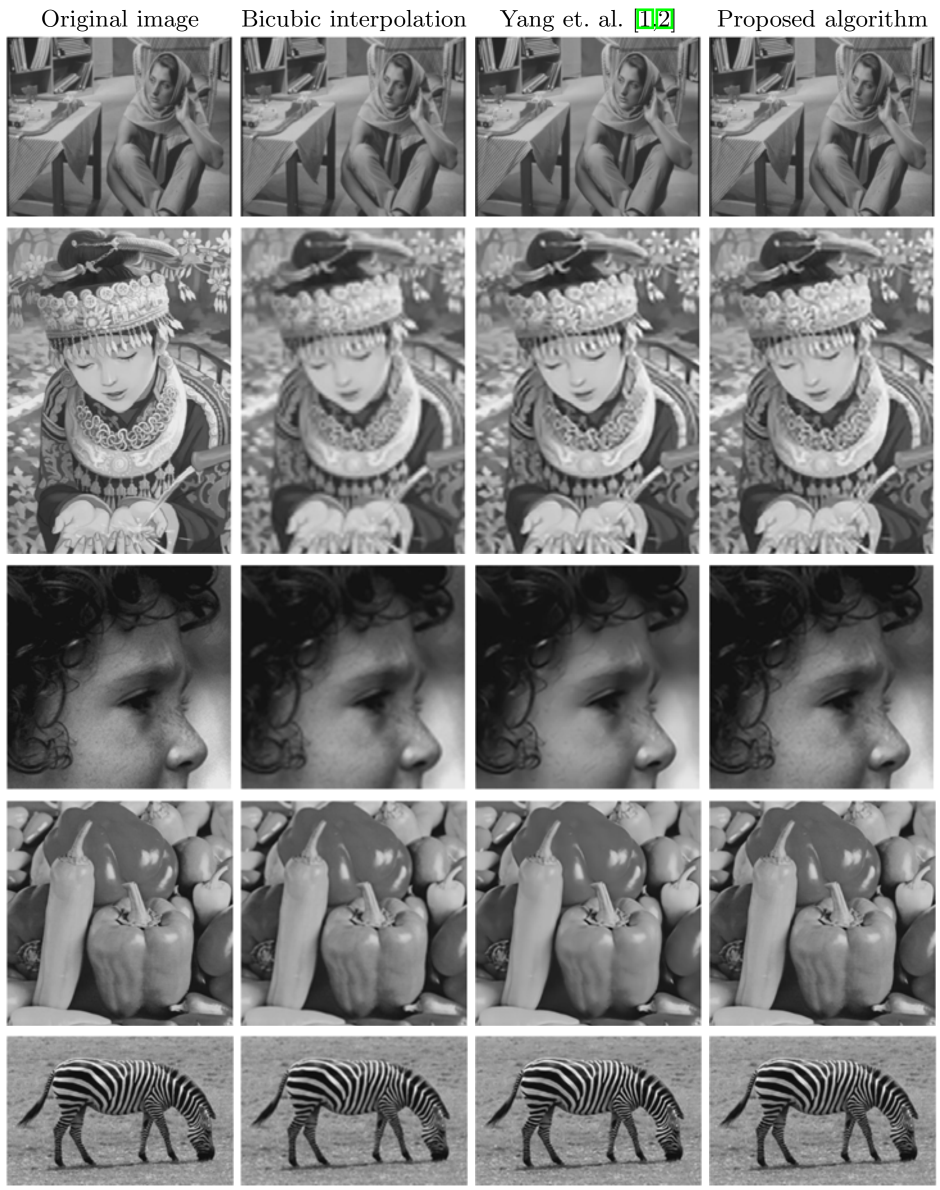


Fig. 5. Visual comparison: Portions from various images (from top to bottom: `barbara`, `comic`, `face`, `pepper`, `zebra`). Left to right: the original image, bicubic interpolation, Yang et. al [1,2] and the proposed algorithm (using the more complex method for dictionary update). Note that the proposed algorithm produces sharper results, preserves the small details of the image and has less visual artifacts compared with bicubic interpolation and Yang et. al. [1,2].

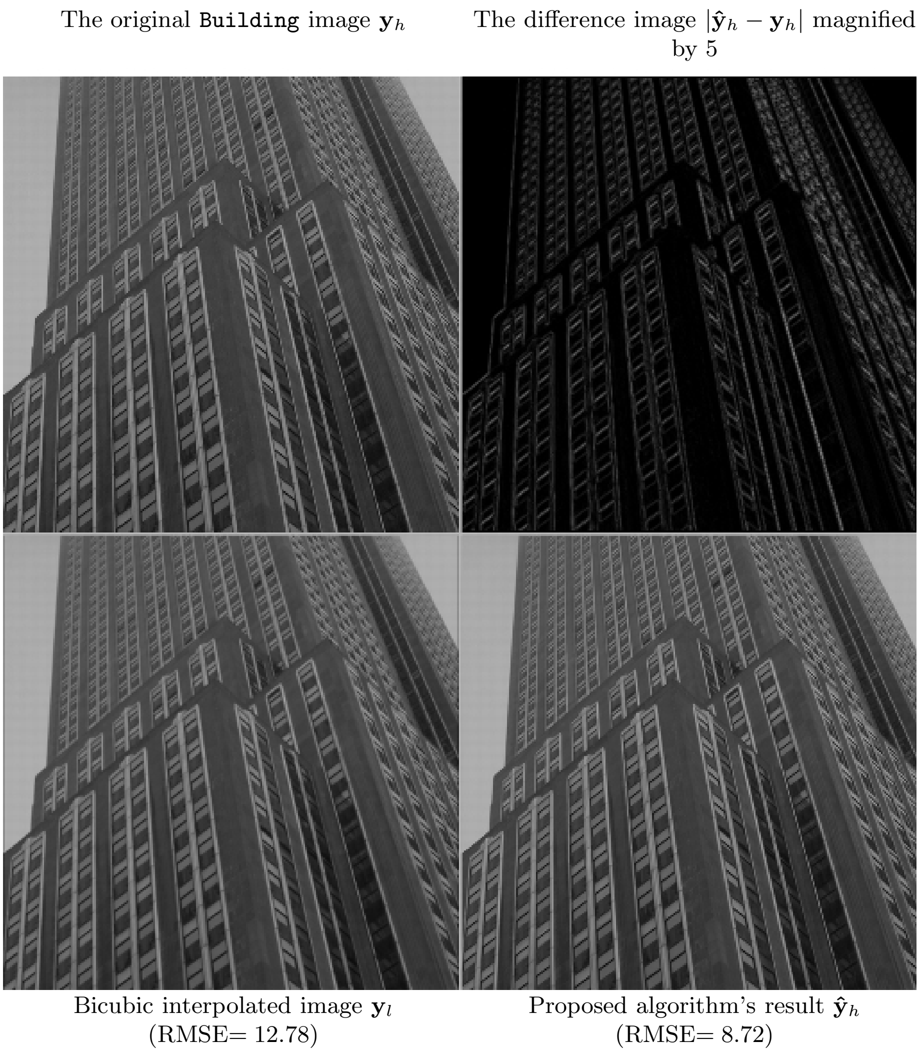

The original Building image $\mathbf{y}_h$
The difference image $|\hat{\mathbf{y}}_h - \mathbf{y}_h|$ magnified by 5
Bicubic interpolated image $\mathbf{y}_l$ (RMSE= 12.78)
Proposed algorithm's result $\hat{\mathbf{y}}_h$ (RMSE= 8.72)

Fig. 6. Bootstrapping experiment

3.32dB, and in order to see where these differences reside, the figure also shows the difference image $|\hat{\mathbf{y}}_h - \mathbf{y}_h|$. Figure 7 shows two 100×100 portions extracted from $\mathbf{y}_h$, $\mathbf{y}_l$, and $\hat{\mathbf{y}}_h$, to better demonstrate the visual gain achieved by the scaling-up algorithm.

This approach has been tested on several images using various dictionary sizes, and in many cases the bootstrapped results are visually comparable and even better, compared to a separate off-line training.

The proposed algorithm results are visually comparable to [22], as demonstrated in Figure 8. Since [22] provides no "ground-truth" images, it is not possible to provide PSNR results. It should be noted that the algorithm of [22] is

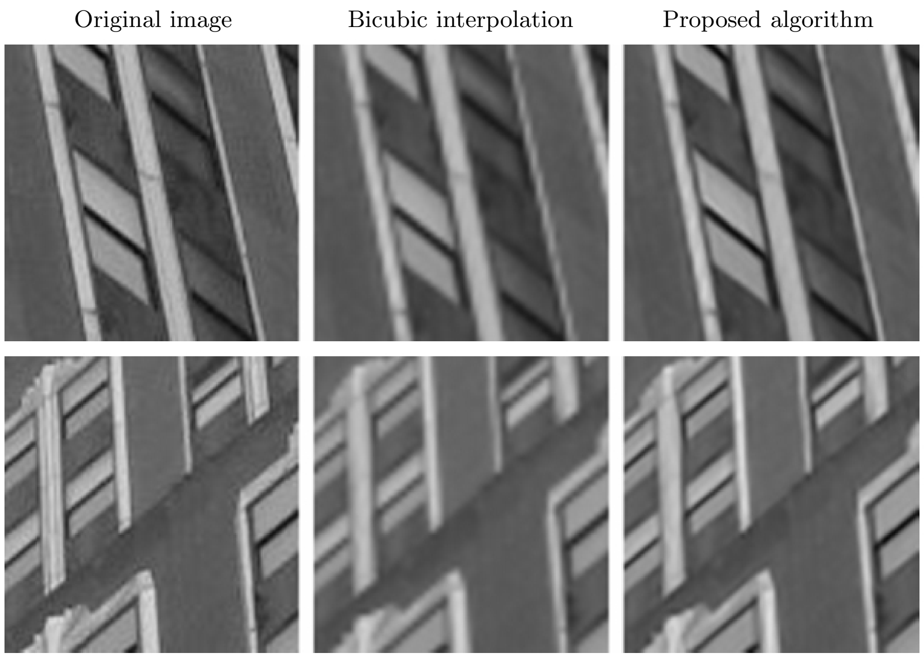


Fig. 7. Bootstrapping experiment: Portions from the `Building` image. Notice that the original portions show some compression artifacts, which do not appear in the scaled-up results

also much more computationally demanding than the proposed one (requiring the solution of many nearest-neighbor problems for the reconstruction phase) and relies heavily on patch recurrence property. Nevertheless, it does performs quite well on the testing set of images described in [22], assumably due to the "coarse-to-fine" approach (where the image is scaled-up n times, each time by $\sqrt[n]{s}$ factor), especially for large scaling factor s.

When compared to the algorithms [1, 2] and [16], the proposed algorithm uses the same idea of training phase and reconstruction phase, Sparse-land modeling of the desired image patches, and a pair of dictionaries that are used to migrate from the low-resolution domain to the high-resolution one. However, different algorithms are used for the dictionary training: K-SVD for the low-resolution dictionary, and pseudo-inverse for the high-resolution dictionary. Moreover, OMP is used for sparse-coding, instead of LASSO optimization methods. Other important modifications in our algorithm are (i) the ability to train on the given image, (ii) the initial interpolation by $\mathbf{Q}$ that simplifies much of the subsequent work without a computational cost, (iii) the definition of the high-resolution patches based on the difference $\mathbf{y}_h - \mathbf{y}_l$, and (iv) the dimensionality reduction we apply on the low-resolution patches.

It should be noted that the sparsity constraint $L = 3$ is used, which is much smaller than the sparsity achieved by [1,2], while the proposed algorithm's PSNR results are better. This may be explained by the better generalization ability of the proposed model, requiring much less atoms per patch. Moreover, while [1,2] suggests training high-resolution and low-resolution dictionaries together (by concatenating high-resolution and low-resolution patches together into one vector), the proposed process is split as described above - thus achieving a more stable reconstruction process having less visual artifacts.

In [1,2] and in [22] it was observed that the result $\hat{\mathbf{y}}_h$ does not necessarily conform with the requirement $\mathbf{L}_{all}\hat{\mathbf{y}}_h = \mathbf{y}_l$. A back-projection procedure was suggested in which the result $\hat{\mathbf{y}}_h$ is projected onto this constraint, by solving the following optimization problem:

$$\hat{\mathbf{y}}_h = \underset{\hat{\mathbf{y}}_h}{\text{argmin}} \|\hat{\mathbf{y}}_h - \mathbf{y}_h\|_2 \ \text{ s.t. } \mathbf{L}_{all}\hat{\mathbf{y}}_h = \mathbf{y}_l. \tag{19}$$

However, our tests show that such a projection procedure is not needed, and in fact, it may add some artifacts to the image. Thus, our algorithm does not use this post-processing stage.

5 Summary

There are various ways to scale-up an image while preserving edges and small details. In this paper we introduced one such algorithm that illustrates how sparse representation modeling and dictionary learning can be used for this goal. The algorithm operates by training a pair of low- and high-resolution dictionaries, using either training images or exploiting a lower-resolution version of the same image to be handled. The presented algorithm is based on the method proposed by Yang et. al. [1,2], with several important modifications:

– Numerical shortcuts bring the proposed algorithm to be highly efficient and much faster (using interpolation of the low-resolution image and dimensionality reduction via PCA).
– A different training approach is used for the dictionary-pair: K-SVD for learning $\mathbf{A}_l$ from extracted features, and direct-approach (using pseudo-inverse) for $\mathbf{A}_h$ from error patches.
– The OMP algorithm is used as sparse coding algorithm, which is much faster than ℓ_1-optimization-based methods.
– The proposed algorithm is much simplified by removing redundant steps (e.g. back-projection during post-processing stage).
– The algorithm can operate without a training-set, by boot-strapping the scale-up task from the given low-resolution image. This idea is similar in spirit to the concept posed in [22], but the proposed solution is simpler and yet very effective.

This method is relatively simple, and yet produces a substantial improvement over bicubic interpolation.

Various further improvements can be considered, and these are likely to improve the algorithm's output quality:

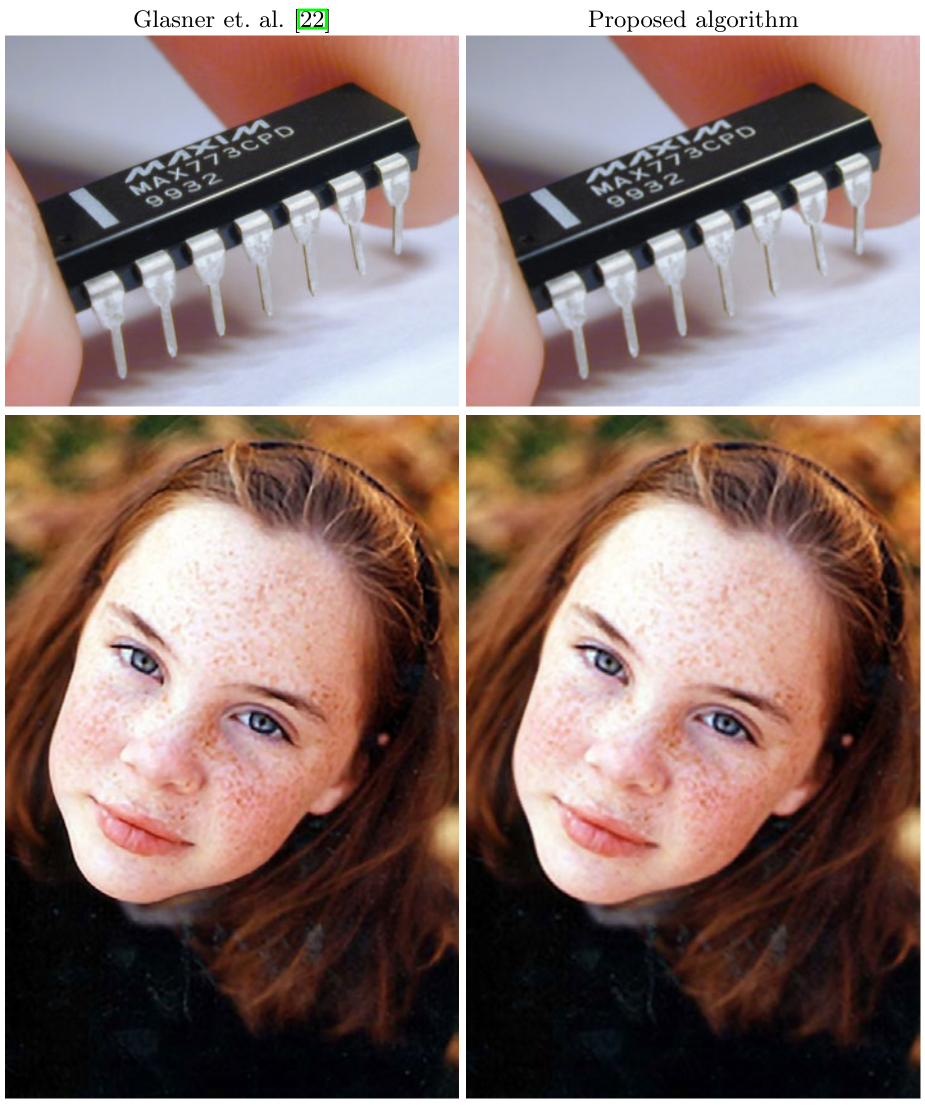


Fig. 8. Visual comparison of the results. First row: Scale-Up by ×4; Second row: Scale-Up by ×3

– It is possible to force the overlapping patches $\hat{\mathbf{p}}_h^k$ to better align with each other. This can be done by operating sequentially on the incoming patches $\mathbf{p}_l^k$, and when applying the sparse coding stage (to produce $\mathbf{q}^k$), a penalty can be added on the distance between the newly constructed patch, $\hat{\mathbf{p}}_h^k$, and the ones already computed. This has been done in [1,2] and quantifying the benefit of this idea should be done.

- Optimization of the feature extraction and dimensionality reduction operators. Various high-pass filters and thresholds have been tested for the PCA stage, however there must be more options to investigate and perhaps even automatically learned.
- The training set can be extended by adding more examples, by applying simple operators on the input images, e.g. rotation by 90°, reflection, etc..
- It should be noted that it is assumed that the blur operator $\mathbf{H}$ is known for all the experiments that have been performed. In the case it is not known (i.e. while bootstrapping from a single image) there is a significant degree of freedom in choosing $\mathbf{H}$, which obviously will affect the results.
- It is possible to combine the off-line training with the bootstrapping approach by training a general dictionary pair $\{\mathbf{A}_l, \mathbf{A}_h\}$ and applying several more iterations on each new test low-resolution image and its down-scaled version $\{\mathbf{z}_l, \mathbf{z}_{ll}\}$. This two stage process will allow the dictionary to "adapt" the reconstruction process to the specific image to be reconstructed.
- Using more than two scales (the "low" and the "high" ones) in a "coarse-to-fine" framework, as practiced in [22] may help improve the scale-up process by building multi-scale sparse-representation for the image.

Acknowledgements. The authors wish to thank the authors of [1,2] for generously sharing their code and data with them. This work was supported in part by the Israel Science Foundation under Grants 1081/07 and 599/08, and by the European Commission's FP7-FET program, SMALL project (grant agreement no. 225913).

References

1. Yang, J., Wright, J., Huang, T., Ma, Y.: Image super-resolution as sparse representation of raw image patches. In: IEEE Computer Vision and Pattern Recognition (CVPR) (June 2008)
2. Yang, J., Wright, J., Huang, T., Ma, Y.: Image super-resolution via sparse representation. IEEE Trans. on Image Processing (to appear)
3. Farsiu, S., Robinson, D., Elad, M., Milanfar, P.: Advances and Challenges in Super-Resolution. International Journal of Imaging Systems and Technology 14(2), 47–57 (2004); special issue on high-resolution image reconstruction
4. Hou, H.S., Andrews, H.C.: Cubic spline for image interpolation and digital filtering. IEEE Transactions on Signal Processing 26, 508–517 (1978)
5. Irani, M., Peleg, S.: Improving Resolution by Image Registration. CVGIP: Graphical Models and Image Processing 53, 231–239 (1991)
6. Schultz, R.R., Stevenson, R.L.: A Bayesian Approach to Image Expansion for Improved Definition. IEEE Transactions on Image Processing 3(3), 233–242 (1994)
7. Freeman, W.T., Pasztor, E.C., Carmichael, O.T.: Learning Low-Level Vision. International Journal of Computer Vision 40(1), 25–47 (2000)
8. Li, X., Orchard, M.: New Edge-Directed Interpolation. IEEE Transactions on Image Processing 10, 1521–1527 (2001)
9. Chang, H., Yeung, D.-Y., Xiong, Y.: Super-resolution through neighbor embedding. In: IEEE Conference on Computer Vision and Pattern Classification (CVPR), vol. 1, pp. 275–282 (2004)

10. Elad, M., Datsenko, D.: Example-Based Regularization Deployed to Super-Resolution Reconstruction of a Single Image. The Computer Journal 50(4), 1–16 (2007)
11. Sun, J., Xu, Z., Shum, H.: Image super-resolution using gradient profile prior. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–8 (2008)
12. Elad, M., Aharon, M.: Image denoising via learned dictionaries and sparse representation. In: International Conference on Computer Vision and Pattern Recognition, New York, June 17-22 (2006)
13. Elad, M., Aharon, M.: Image denoising via sparse and redundant representations over learned dictionaries. IEEE Trans. on Image Processing 15(12), 3736–3745 (2006)
14. Bruckstein, A.M., Donoho, D.L., Elad, M.: From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images. SIAM Review 51(1), 34–81 (2009)
15. Elad, M.: Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing. Springer, Heidelberg (2010)
16. Wang, J., Zhu, S., Gong, Y.: Resolution enhancement based on learning the sparse association of image patches. Pattern Recognition Letters 31(1) (January 2010)
17. Lou, Y., Bertozzi, A., Soatto, S.: Direct sparse deblurring. J. Math. Imag. Vis. (August 13, 2010)
18. Zhang, X., Wu, X.: Image interpolation by adaptive 2-d autoregressive modeling and soft-decision estimation. IEEE Trans. Image Process. 17(6), 887–896 (2008)
19. Mallat, S., Yu, G.: Super-Resolution with Sparse Mixing Estimators. IEEE Trans. on Image Processing 19(11), 2889–2900 (2010)
20. Aharon, M., Elad, M., Bruckstein, A.M.: The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation. IEEE Trans. on Signal Processing 54(11), 4311–4322 (2006)
21. Rubinstein, R., Zibulevsky, M., Elad, M.: Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit, Technical Report - CS Technion (April 2008)
22. Glasner, D., Bagon, S., Irani, M.: Super-Resolution from a Single Image. In: International Conference on Computer Vision, ICCV (October 2009)

Periodic T-Splines and Tubular Surface Fitting

Jianmin Zheng[1] and Yimin Wang[2]

[1] School of Computer Engineering, Nanyang Technological University, Singapore
[2] Next Generation Web, IBM Research, Beijing 100193, China

Abstract. This paper discusses a special type of T-spline surfaces called periodic T-splines that are closed in one parameter direction, and their application in tubular surface fitting. First, a global representation is proposed for representing periodic T-splines. This representation does not require repeating control points, which facilitates surface fitting process. Then, an algorithm for adaptively fitting periodic T-splines to a tubular triangular mesh that has the same topology as a cylinder is presented. The resulting periodic T-spline is obtained respecting the geometric distribution of the input mesh. The use of periodic T-splines for tubular surface fitting has at least two advantages: 1) adaptive fitting is easily achieved due to the local refinement of T-splines; 2) the algorithm avoids cutting the mesh to make it a disk topologically for conventional B-spline fitting due to the periodic representation and this overcomes the drawback of finding a good cutting path, which is usually difficult.

Keywords: Periodic T-splines, tubular surfaces, adaptive surface fitting.

1 Introduction

Triangular meshes and splines are two typical representations for freeform surfaces. In some applications, there is a need to convert a model in triangular mesh representation into splines. This conversion can be achieved by surface fitting or reconstruction. Many interesting methods have been developed for reconstructing spline surfaces from scattered data or meshes [1,2,3,4,5].

This paper considers the problem of converting a tubular triangular mesh into a spline surface, as depicted in Figure 1 where the left is an input tubular mesh and the right is an output spline surface. A tubular surface is characterized by having two boundary loops and one lateral surface between them [6,7]. The lateral surface is open along one direction (the axis direction) and closed along the other direction (the sectional direction). A tubular surface can be viewed as a deformation of a cylinder. The two loops correspond to the upper and lower boundaries of the cylinder and the lateral surface corresponds to the interior of the cylinder surface (see Figure 2). Tubular surfaces are still quite common in many application domains. For example, pipes in mechanical engineering and blood vessels in medical engineering are tubular surfaces. Although previous fitting methods can be used to fit spline surface to a tubular mesh, it is required to cut or segment the tubular mesh into a disk or disks topologically. Finding a good cutting or segmentation is usually difficult.

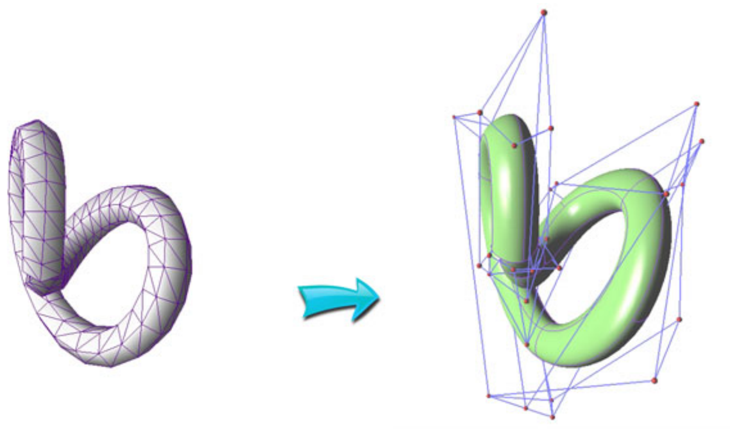


Fig. 1. Converting a tubular mesh (left) to a spline surface (right)

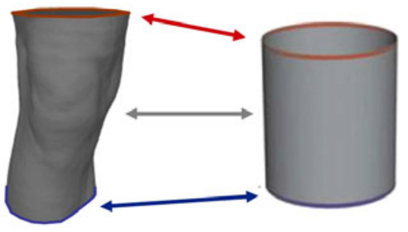


Fig. 2. Correspondence between a tubular surface (left) and a cylinder (right)

Alternatively, in this paper we propose to use periodic T-spline surfaces to approximate tubular meshes. We choose T-splines because T-splines allow us to easily implement adaptive fitting. In practice, adaptive fitting is attractive. It handles complicated models with unevenly distributed details well. Since tensor-product B-spline surfaces cannot be refined locally, hierarchical B-splines were used to perform adaptive fitting [4,5]. However, in hierarchical B-splines, hierarchy must be taken care of. T-splines are a recently-developed free-form surface technology that solves most of the limitations inherent in the non-uniform rational B-spline (NURBS) representation [8,9]. T-splines allow local refinement, making them a good candidate for adaptive fitting [10,11]. T-splines are forward and backward compatible with NURBS and thus integrate well into existing CAD/CAM systems.

We choose periodic surfaces because tubular surfaces are closed in one direction. Therefore we are interested in a class of T-splines that are closed in one parameter direction. Such periodic T-splines can be viewed as a generalization of one-directionally periodic B-spline surfaces by permitting the existence of T-junction control points in their control grids, thus enabling local refinement.

They can seamlessly model those surfaces that have the same topology as a cylinder surface.

To accomplish adaptive fitting using periodic T-splines, we examine two basic tasks in this paper. The first task is how to appropriately represent periodic T-splines. A global representation for periodic T-splines is proposed in Section 2, which facilitates T-spline fitting process. The second task is a fitting algorithm for converting a mesh represented tubular surface into a periodic T-spline surface. The detailed steps of our adaptive fitting algorithm are described in Section 3. Two experimental examples are provided in Section 4 and Section 5 concludes the paper.

2 Periodic T-Splines

In CAGD, periodic curves and surfaces often refer to those curves and surfaces that are closed, seemingly without beginning or end [12]. In this section, we first explain why the common process in CAGD for handling periodic B-splines may not be convenient for periodic T-spline fitting and then propose an alternative approach to handling periodic T-splines. To simplify the discussion, this paper is written in terms of cubic B-spline curves and bicubic T-splines. The approaches can be extended to arbitrary degree B-splines/T-splines.

2.1 Repeating Control Points and Knot Intervals for Periodic B-Splines

A common approach in CAGD for constructing a periodic B-spline curve is to treat the periodic curve as a simple special case of an open curve [12]. The open curve needs some extra control points that overlap the existing ones. We explain this with an example. Figure 3(a) shows four points forming a polygon. If we want to define a closed cubic B-spline curve as shown in Figure 3(b), we need more control points to make the curve consist of several segments. In Figure 3(c), three control points are added, which makes the shape of the curve is close to that of Figure 3(b) but not "quite" periodic. If the three new control points repeat the first three of the input points and the knot intervals for the two new edges are the same as those for the first two edges, then the curve becomes periodic as shown in Figure 3(d).

The idea of repeating control points and knot intervals can be extended to B-spline surfaces. Figure 4 is such an example, where (a) and (b) show the control mesh in 3D and the parameter domain, respectively, and (c) shows the repeating of control points and knot intervals.

The advantage of the above repeating approach is that periodic B-splines can be treated as the "normal" ones and there is no need to write special B-spline routines for the periodic case. However, the disadvantages are also obvious, especially for surface fitting where the control points are to be determined. First, the control point repeating approach imposes constraints on some control points. Thus not all the control points are treated equally. Second, when the approach is

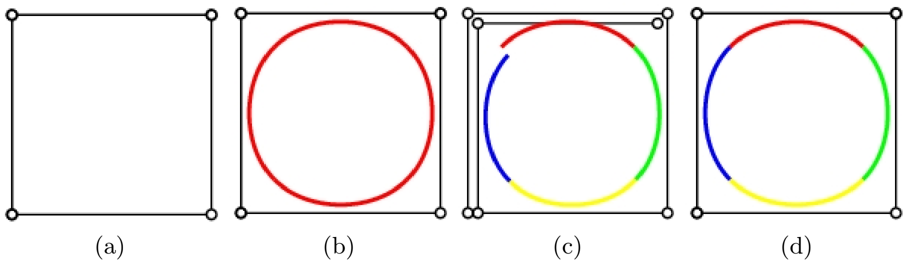


Fig. 3. Repeating control points for a periodic B-spline curve

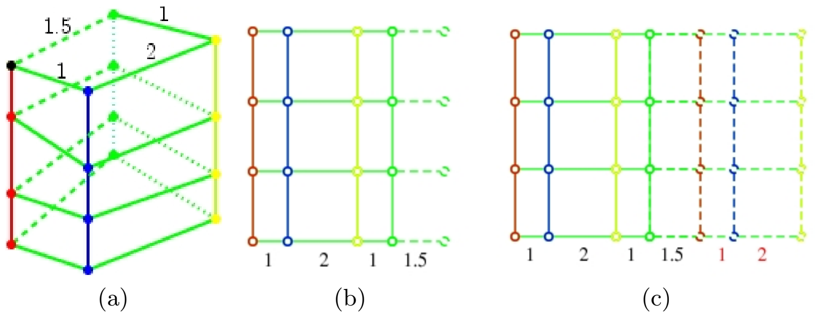


Fig. 4. Repeating control points for a periodic B-spline surface

applied to a T-spline, the situation becomes much more complicated because it may have many different repeating patterns due to the flexibility of the T-spline control grid. Refer to the example in Figure 5. The T-spline control points of four rows that are unfolded onto the parameter domain are shown in Figure 5(a), where the black dash lines are the edges completing the polygons. Without ambiguity, we do not display the vertical edges in the T-mesh for clarity. If we choose the second point of the lowest row to be the local origin with the horizontal parameter value of 0, then we need add three control points on the right, which are the same as the first three control points on the left, as depicted in Figure 5(b). The range of the horizontal parameter is from 0 to 4 and the two vertical dash lines are drawn to indicate the parameter borders. With this parameter range of [0,4], the top three rows shall have different repeating patterns of the control points, as illustrated in Figure 5(c).

2.2 New Construction for Periodic B-Splines

We have seen in the preceding subsection that the approach of repeating control points and knot intervals for handling periodic B-splines is not convenient for our periodic T-spline fitting application. This motivates us to search for other approaches. In this subsection, we describe another way to handle periodic B-splines. The basic idea is to modify the basis functions, instead of repeating control points.

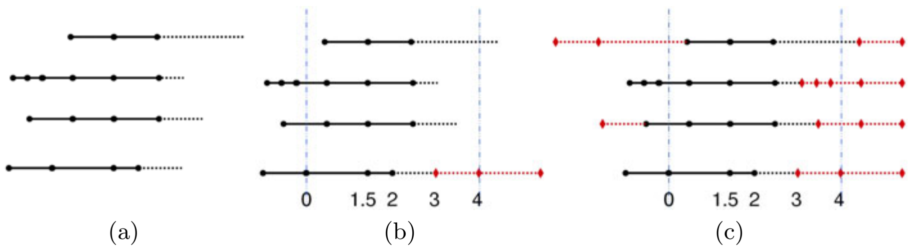
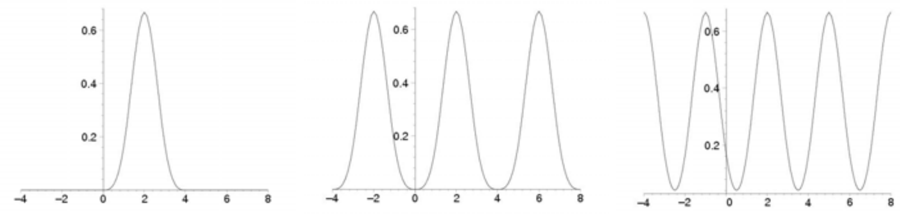


Fig. 5. Different repeating patterns for a periodic T-spline



(a) $N[0,1,2,3,4](t)$ (b) $\tilde{N}_4[0,1,2,3,4](t)$ (c) $\tilde{N}_3[0,1,2,3,4](t)$

Fig. 6. One B-spline basis function $N[0,1,2,3,4](t)$ and two periodic B-spline functions $\tilde{N}_4[0,1,2,3,4](t)$ and $\tilde{N}_3[0,1,2,3,4](t)$

Note that B-spline basis functions are piecewise polynomials with finite support and by nature they are not periodic. To deal with closed shapes seamlessly, it seems more appropriate to use periodic functions. In the following we construct periodic blending functions from B-spline basis functions.

Consider the cubic B-spline basis function $N[\mathbf{t}_i](t)$ associated with knot vector $\mathbf{t}_i = [t_{i-2}, t_{i-1}, t_i, t_{i+1}, t_{i+2}]$. It is a piecewise cubic polynomial within $[t_{i-2}, t_{i+2}]$ and vanishes for t outside of $[t_{i-2}, t_{i+2}]$. Given a constant T as a period, we define

$$\tilde{N}_T[\mathbf{t}_i](t) = \sum_{j=-\infty}^{+\infty} N[\mathbf{t}_i](t + jT) = \sum_{j=-\infty}^{+\infty} N[\mathbf{t}_i - jT](t) \tag{1}$$

where $N[\mathbf{t}_i](t + jT) = N[\mathbf{t}_i - jT](t)$ is also a B-spline basis function associated with knot vector $[t_{i-2} - jT, t_{i-1} - jT, t_i - jT, t_{i+1} - jT, t_{i+2} - jT]$. Obviously, $\tilde{N}_T[\mathbf{t}_i](t)$ is a periodic function with period T. When $T \geq t_{i+2} - t_{i-2}$, $\tilde{N}_T[\mathbf{t}_i](t)$ coincides with $N[\mathbf{t}_i - jT](t)$ in $[t_{i-2} - jT, t_{i+2} - jT]$ for any integer j. Refer to Figure 6 for an illustration. $N[0,1,2,3,4](t)$ is the same as $\tilde{N}_4[0,1,2,3,4](t)$ in $[0,4]$. However, when $T = 3$, $N[0,1,2,3,4](t)$ and $\tilde{N}_3[0,1,2,3,4](t)$ are different.

With the periodic function $\tilde{N}_T[\mathbf{t}_i](t)$, it is easy to derive an explicit representation for a periodic B-spline curve. For example, given $n+1$ points $P_0, P_1, \cdots, P_n$ forming a closed control polygon and $n+2$ knots $t_0, t_1, \cdots, t_{n+1}$ to define a periodic cubic B-spline curve, we define the period $T = t_{n+1} - t_0$. Then the periodic B-spline curve can be represented by

$$C(t) = \sum_{i=0}^{n} P_i \tilde{N}_T[\mathbf{t}_i](t), \qquad t \in (-\infty, +\infty).$$

Obviously, for any t, $C(t+T) = C(t)$. That is, the shape of the curve is completely determined by $C(t)$ within a finite parameter interval $[a, a+T)$ for any number a. In practice, we are always interested in the curve within the domain $[t_0, t_{n+1}]$.

It is interesting to note that though $\tilde{N}_T[\mathbf{t}_i](t)$ is a sum of an infinite number of B-spline basis functions, there are only a finite number of terms that do not vanish in domain $[t_0, t_{n+1})$. In fact, consider $\tilde{N}_T[\mathbf{t}_i](t)$ corresponding to the control point P_i. When the number of the control points is greater than 1 (i.e., $n > 0$), we have $t_{i+2} - t_i \le T$ and $t_i - t_{i-2} \le T$ for $\mathbf{t}_i = [t_{i-2}, t_{i-1}, t_i, t_{i+1}, t_{i+2}]$. Then all $N[\mathbf{t}_i - jT](t)$ and $N[\mathbf{t}_i + jT](t)$ vanish in $[t_0, t_{n+1}]$ for $j > 1$.

Periodic NURBS surfaces can be defined in a similar way.

2.3 Periodic T-Spline Surface Representation

With the periodic function $\tilde{N}_T[\mathbf{t}_i](t)$, we can also globally represent periodic T-spline surfaces that are periodic in one parameter direction (say u). Such periodic T-spline surfaces are a generalization of periodic tensor-product NURBS surfaces, the main difference being that periodic T-splines permit partial rows or columns of control points. The terminal control point in a partial row or column is called a T-junction. Similar to T-splines [8,9], a control grid for a periodic T-spline surface is called a T-mesh. All the rules applied to T-splines are also valid for periodic T-splines. The T-mesh of a periodic T-spline is closed in one direction. Figure 7 shows a T-mesh and its periodic T-spline surface. If a T-mesh happens to form a grid with no T-junctions, the corresponding periodic T-spline surface degenerates to a periodic NURBS surface.

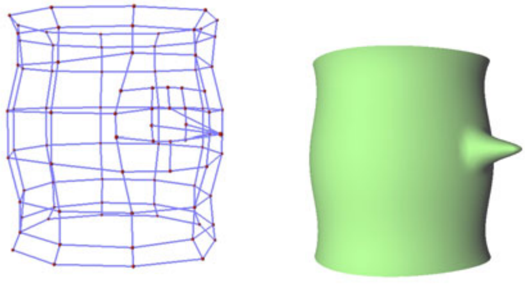


Fig. 7. A T-mesh and its periodic T-spline surface

Knot information for periodic T-splines is given by knot intervals assigned to each edge of the T-mesh. Figure 8 shows the pre-image of a T-mesh in (u, v) parameter space. Since the T-mesh is closed in the u direction, we arbitrarily choose a column (for example, c_1) as a virtual border and unfold the T-mesh along the virtual border. Column c_2 is a virtual duplicate of column c_1. P_1 and P_1' actually correspond to the same control point in the T-mesh. Both c_1 and c_2 are called the left and right virtual borders. d_i and e_j denote the knot intervals. Knot intervals are constrained by the relationship that the sum of all knot intervals along one side of any face must equal the sum of the knot intervals on the opposite side [8,9]. For example, for face F in Figure 8, $d_1 + d_6 = d_9$ and $e_6 + e_7 = e_9 + e_{10}$. It can also be seen that the sum of the knot intervals in row r_1 equals the sum of the knot intervals in row r_2. Thus this sum is used as the period T for the u direction.

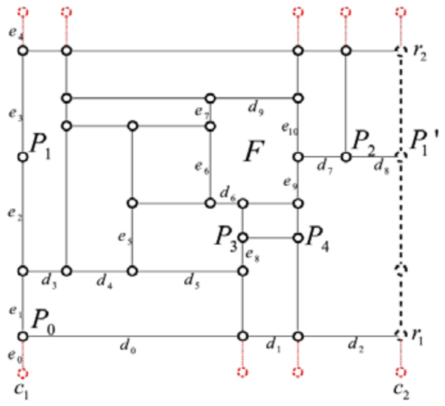


Fig. 8. Pre-image for a periodic T-spline surface

To introduce knots, we impose a local knot coordinate system on the T-mesh. This can be done by arbitrarily choosing a control point whose pre-image (say P_0 in Figure 8) serve as the origin for the parameter domain $(u, v) = (0, 0)$ and then assigning a u knot value to each vertical edge and a v value to each horizontal edge based on the knot interval information. After that, each control point has knot coordinates. For example, P_1, P_2 and P_3 have knot coordinates $(0, e_1 + e_2)$, $(d_0 + d_1 + d_7, e_1 + e_2)$ and $(d_0, e_1 + e_8)$, respectively.

With the local knot coordinate system and period T, we can write an explicit formula for the periodic T-spline surface:

$$\mathbf{S}(u, v) = \frac{\sum_{i=0}^{n} w_i P_i B_i^*(u, v)}{\sum_{i=0}^{n} w_i B_i^*(u, v)} \tag{2}$$

where $P_i = (x_i, y_i, z_i)$ are control points in 3D space, w_i are control point weights, n is the number of the control points, and $B_i^*(u, v)$ are periodic T-spline blending functions which are given by:

$$B_i^*(u, v) = \tilde{N}_T[\mathbf{u}_i](u) \cdot N[\mathbf{v}_i](v).$$

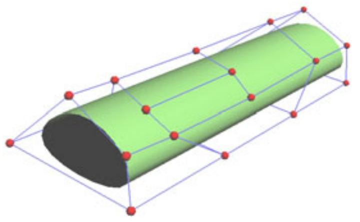
The knot vectors $\mathbf{u}_i = [u_{i0}, u_{i1}, u_{i2}, u_{i3}, u_{i4}]$ and $\mathbf{v}_i = [v_{i0}, v_{i1}, v_{i2}, v_{i3}, v_{i4}]$, which are used to define the cubic B-spline basis functions $\tilde{N}_T[\mathbf{u}_i](u)$ and $N[\mathbf{v}_i](v)$, are inferred from the T-mesh. The rule used in [8,9] are adapted to handle the periodic case: Assume (u_{i2}, v_{i2}) are the knot coordinates for P_i. To find u_{i3} and u_{i4}, we cast a ray from P_i in the parameter domain: $R(t) = (u_{i2} + t, v_{i2}),\ t > 0$. Then u_{i3} and u_{i4} are defined as the u coordinates of the first two vertical edges intersected by the ray $R(t)$. Note that when the ray crosses the right virtual border, it continues from the left virtual border and the period T should be added to the knot obtained later. Knots u_{i0} and u_{i1} in $\mathbf{u}_i$ are found likewise. Thus, the u and v knot vectors for P_1 are $[-(d_7 + d_8), -d_8, 0, d_3, d_3 + d_4]$ and $[0, e_1, e_1 + e_2, e_1 + e_2 + e_3, e_1 + e_2 + e_3 + e_4]$. The u and v knot vector for P_3 are $[d_3, d_3 + d_4, d_0, d_0 + d_1, d_0 + d_1 + d_2]$ and $[0, e_1, e_1 + e_8, e_1 + e_5, e_1 + e_5 + e_6 + e_7]$.

It is worth pointing out that the T-mesh of a periodic T-spline surface could have quite flexible structure as long as it is topologically equivalent to a cylinder. For example, it could have an arbitrary number of control points along one periodic row and there is no need for a complete vertical connection in the T-mesh.

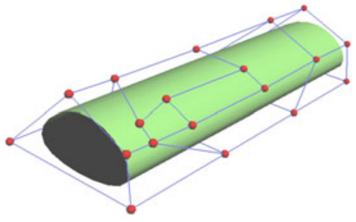
Control Point Insertion. One important feature of T-splines is local refinement. Periodic T-spline surfaces have the same property. The main idea of the T-spline local refinement algorithm [9] is to maintain the validity of the T-mesh and to ensure that the B-spline basis functions and the control points are properly associated. In the periodic T-spline case, each control point may correspond to several B-spline basis functions. Therefore the original T-spline local refinement algorithm could be adapted to ensure that each control point is properly associated to each corresponding B-spline basis function. Figure 9 shows an example of local knot insertion, where (a) is a periodic T-spline surface together with its T-mesh and (b) is the same periodic T-spline surface but the T-mesh changes due to the local insertion of two points.

3 Periodic T-Spline Surface Fitting

Now we discuss how to automatically convert a tubular mesh into a periodic T-spline surface. As shown in Figure 1, the input is a tubular mesh, comprised of a number of triangle faces and the output is a smooth periodic T-spline surface that approximates the tubular mesh. Mathematically, this problem can be formulated as follows: given a tubular triangular mesh $\mathbf{M}$ which is defined by a set of vertices $V = \{\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_m\}$ and their connectivity information, we want to find a periodic T-spline surface $\mathbf{S}(u, v)$ that approximates each vertex of the tubular mesh within an error tolerance ε, i.e. $\text{dist}(\mathbf{d}_i, \mathbf{S}(u, v)) \le \varepsilon,\ i = 1, 2, \cdots, m$.



(a) A periodic T-spline surface and its T-mesh



(b) The same periodic T-spline surface and its new T-mesh after local insertion of two points

Fig. 9. Control point insertion for a periodic T-spline surface

The process involves determining the pre-image of the T-mesh, weights and geometry of the control points.

Here we present an adaptive algorithm to find the periodic T-spline surface. The algorithm consists of several steps (see Figure 10): parameterization, initial placement of the T-mesh, geometry optimization, and error check/T-mesh refinement. The geometry optimization and T-mesh refinement are carried out iteratively to improve the fitting result until the approximation error is within the prescribed tolerance. Since this framework is quite standard for adaptive fitting, our description in the following will emphasize special technical components due to the use of periodic T-splines.

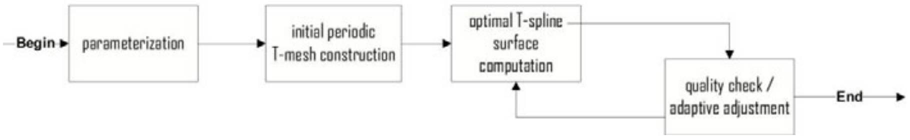

Begin
parameterization
initial periodic T-mesh construction
optimal T-spline surface computation
quality check / adaptive adjustment
End

Fig. 10. The flowchart of periodic T-spline surface fitting

3.1 Parameterization

Parameterization is a process of assigning a pair of parameter values to each vertex of the input tubular mesh. Since it establishes a mapping between the tubular mesh and periodic T-spline surfaces, the quality of parameterization affects the fitting results. Many parameterization methods have been proposed, but most of them assume that the mesh is an open mesh. In our algorithm, we use an edge based parameterization method [7], which is designed specifically for parameterizing tubular meshes. In this method, the edges rather than the vertices are treated as the target for parameterization. As a result, the method does not need to cut the mesh into a disk topologically. It thus overcomes the problems of cutting paths that are the zigzag paths leading to suboptimal parameterizations and the difficulty in finding good cutting paths.

3.2 Initial Placement of the T-Mesh

In this step, we construct an initial placement of the T-mesh, with which the adaptive surface fitting process can begin. First, the initial placement of the T-mesh is chosen to be a regular $dim_u \times dim_v$ grid over the domain $[0,1] \times [0,1]$ which corresponds to the result of the parameterization. The knot vectors along the $u-$ and $v-$directions associated with the T-mesh are $\{0, \frac{1}{dim_u-1}, \frac{2}{dim_u-1}, \cdots, \frac{dim_u-2}{dim_u-1}\}$ and $\{-0.02, -0.01, 0, \frac{1}{dim_v-3}, \frac{2}{dim_v-3}, \cdots, 1, 1.01, 1.02\}$, respectively. For simplicity, the sizes of the control grid, dim_u and dim_v, are often set to 4. An example of the initial placement of a 4×4 periodic T-mesh is shown in Figure 11. In the right of the figure, the 4 control points in the rightmost column are virtual duplicates of those in the leftmost column and the domain of the surface is the region inside of the red bounding box.

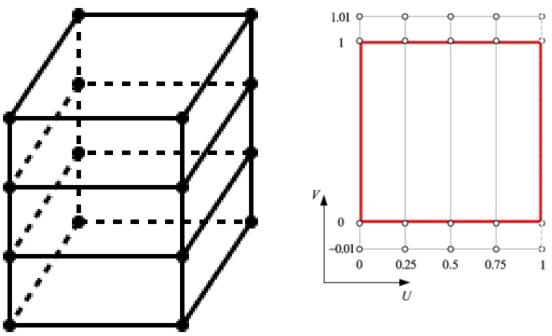


Fig. 11. The initial placement (left) and pre-image (right) of a 4×4 periodic T-mesh

Second, we let all the control point weights w_i be 1. The initial pre-image of the T-mesh and the weights define a class of periodic T-spline surfaces that have the same T-mesh topological structure and the same weights. The geometric coordinates of the control points are to be determined by optimization in the next step.

3.3 Geometry Optimization

Consider the class of periodic T-splines that are determined by the pre-image of the T-mesh and the weights. The pre-image of the T-mesh and the weights are the ones either set in the step of initial placement of the T-mesh or produced by the local refinement of the initial placement in the step of T-mesh refinement. Each individual surface in this class differs from the others only by the geometry of the control points. We seek to find the control points that define a periodic T-spline surface best fitting the tubular mesh in the least-squares meaning.

It is important to note that any periodic T-spline surface in the above class satisfies $\sum_{i=0}^{n} w_i B_i^*(u,v) \equiv 1$. This is because the initial class given in the step of initial placement consists of only bicubic B-splines that are standard T-splines and our local refinement algorithm converts a standard T-spline into either a standard or a semi-standard T-spline [8,9]. Both standard or semi-standard T-splines are piecewise polynomial. Therefore we only need to focus on the numerator:

$$\mathbf{S}(u,v) = \sum_{i=0}^{n} w_i P_i B_i^*(u,v) \tag{3}$$

This simplifies the optimization computation significantly. The optimal control points P_i are determined such as the following objective function is minimized:

$$F(P_0, P_1, \cdots, P_n) = \sum_{j=1}^{m} \parallel \mathbf{S}(u_j, v_j) - \mathbf{d}_j \parallel^2 + \sigma J_{fair}(S) \tag{4}$$

where the first term is fidelity measuring the distances between the vertices of the mesh and the periodic T-spline surface, the second term is a fairness term, and σ is a tradeoff factor that balances the approximation and fairness. There are a few choices for the fairness functions. Considering the low computational complexity, we choose the simple thin plate energy:

$$J_{fair}(\mathbf{S}) = \iint (\mathbf{S}_{uu}^2(u,v) + 2\mathbf{S}_{uv}^2(u,v) + \mathbf{S}_{vv}^2(u,v))dudv \tag{5}$$

where $\mathbf{S}_{uu}$, $\mathbf{S}_{uv}$, $\mathbf{S}_{vv}$ are the second order partial derivatives of $\mathbf{S}(u,v)$.

This is a typical least-squares problem. To find the solution, we differentiate the objective function $F(P_0, P_1, \cdots, P_n)$ with respect to each control point P_k and let the partial derivative equal zero: $\dfrac{\partial F}{\partial P_k} = 0$, which leads to

$$\sum_{i=0}^{n} w_i \left(\sum_{j=1}^{m} B_i^*(u_j, v_j) B_k^*(u_j, v_j) + \sigma m_{ik} \right) P_i = \sum_{j=1}^{m} \mathbf{d_j} B_k^*(u_j, v_j) \tag{6}$$

for $k = 0, \cdots, n$, where m_{ik} is

$$\begin{aligned} m_{ik} = & \int_0^T \frac{d^2\tilde{N}_T[\mathbf{u}_i](u)}{du^2} \cdot \frac{d^2\tilde{N}_T[\mathbf{u}_k](u)}{du^2} du \int_0^1 N[\mathbf{v}_i](v) \cdot N[\mathbf{v}_k](v) dv \\ & +2 \int_0^T \frac{d\tilde{N}_T[\mathbf{u}_i](u)}{du} \cdot \frac{d\tilde{N}_T[\mathbf{u}_k](u)}{du} du \int_0^1 \frac{dN[\mathbf{v}_i](v)}{dv} \cdot \frac{dN[\mathbf{v}_k](v)}{dv} dv \\ & + \int_0^T \tilde{N}_T[\mathbf{u}_i](u) \cdot \tilde{N}_T[\mathbf{u}_k](u) du \int_0^1 \frac{d^2N[\mathbf{v}_i](v)}{dv^2} \cdot \frac{d^2N[\mathbf{v}_k](v)}{dv^2} dv. \end{aligned}$$

The linear system is then solved using some linear solvers such as the preconditioned complex bi-conjugate gradient (PCBCG) solver.

3.4 T-Mesh Refinement

After a periodic T-spline surface is computed under the current class, we need to check whether the approximation meets the requirement. The approximation error, in term of the parametric distance of each vertex $\mathbf{d}_i$ of the mesh to the T-spline surface: $\|\mathbf{S}(u_i, v_i) - \mathbf{d}_i\|$, is evaluated. If all the distances are below the tolerance ε, the T-spline surface is considered acceptable and the algorithm outputs the result. Otherwise, we check each face of the pre-image of the T-mesh in the parameter domain. If it contains the pre-image of violating vertices at which the approximation error is greater than the tolerance, we call it an offending face and it needs to be refined.

We refine each offending face by splitting the face into two halves of equal size. To do this, we simply insert two end points of the edge into the T-mesh using the local refinement method for periodic T-spline surfaces. The edge could be either horizontal or vertical, depending on whether the height or the width of the face is larger.

After the T-mesh is updated, we obtain a new class of periodic T-splines determined by the new setting of the pre-image of the T-mesh and the new control point weights. Since the new T-mesh structure and the control point weights are generated by the T-spline local refinement algorithm, the resulting class of periodic T-spline surfaces remains to be standard or semi-standard. Therefore, surfaces in this class are still polynomial. The control points of the optimal T-spline in this class can be computed using the geometry optimization step described in the preceding subsection.

4 Examples

This section presents two examples to demonstrate the proposed periodic T-spline surface fitting algorithm.

The first example is a bumpy model represented by a tubular triangular mesh (see Figure 12(a)). When we use the fitting algorithm to approximate the bumpy model, the error tolerance for approximation is set to be 0.5% of the size of the model and the fairness factor is set to be 0.00001. Figure 12(b) and Figure 12(c) show two periodic T-spline surfaces which are the intermediate results after 3 and 6 iterations respectively, with the pre-images mapped onto the surfaces. It can be seen that the structures of the T-meshes are not sufficiently sophisticated to represent all the geometrical features of the original tubular mesh and the surfaces are not well approximated. After 10 iterations, a satisfactory result is obtained, which is shown in Figure 12(d) with the pre-image mapped onto the surface (a window is also drawn to highlight three T-junction points in red color illustrating the existence of T-junctions). Figure 12(e) and Figure 12(f) are the final periodic T-spline surface without and with the T-mesh. Refer to Table 1 for

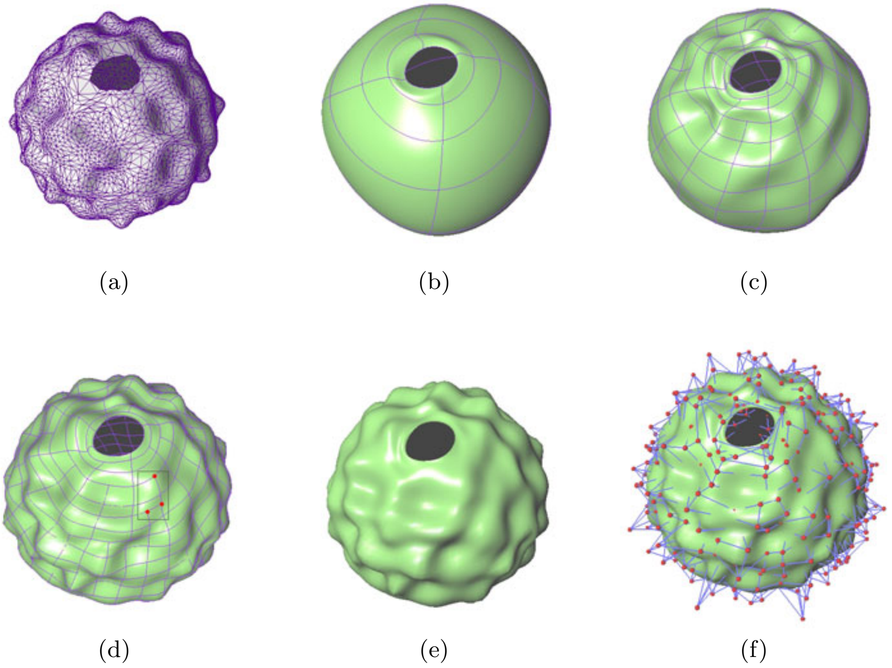


Fig. 12. Adaptively fitting a bumpy model. (a): input model; (b)-(d): results at different stages of fitting; (e) and (f): the final T-spline surface and the control mesh

the detailed information about the input mesh and the final periodic T-spline surface.

The second example is to demonstrate the algorithm in approximating tubular meshes that have relatively low quality (i.e., the number of vertices is insufficient compared to the complexity of the geometry they represent). It can be seen that the algorithm is capable of handling these meshes by choosing a smaller error tolerance. The model we use in the second example is the Venus shown in Figure 13(a), which is open at top and bottom and has 710 vertices and 1389

Table 1. Statistics of the first example

Input mesh	Number of vertices	5610
	Number of faces	11181
Output periodic T-spline surface obtained after 10 iterations	Number of knot in the $u-$direction	33
	Number of knot in the $v-$direction	32
	Number of control points	570
	Number of control points of an equivalent NURBS	924
	Maximum approximation error	0.44%
	Average approximation error	0.03%

Table 2. Statistics of the second example

Input mesh	Number of vertices	710
	Number of faces	1389
Output periodic T-spline surface obtained after 10 iterations	Number of knot in the $u-$direction	20
	Number of knot in the $v-$direction	27
	Number of control points	243
	Number of control points of an equivalent NURBS	460
	Maximum approximation error	0.018%
	Average approximation error	0.001%

faces. If we choose the error tolerance to be 0.5% of the size of the model, the fitting periodic T-spline surface can be quickly obtained. However, the resulting surface (see Figure 13(b)) does not fit the shape of the mesh very well although the error tolerance of 0.5% is already met at all the vertices. This is because when the density of the vertices of the model is not sufficiently high and the approximation tolerance is not sufficiently small, the resulting surface may not approximate the edges or faces of the mesh very well. The resulting surface (shown Figure 13(c)) could be improved if the error tolerance is changed to 0.05% of the size of the model, but it is still not good enough. Finally, if we adjust the error tolerance to be 0.02% of the size of the model, this time we get

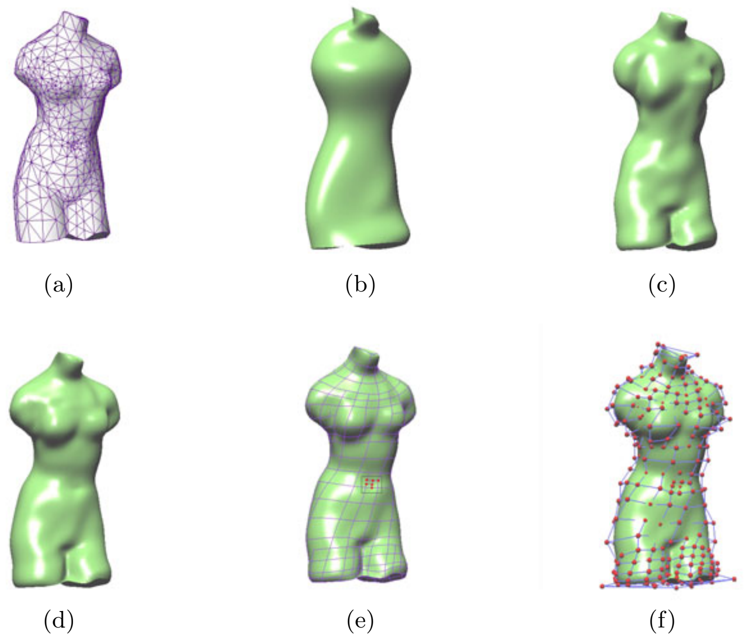


Fig. 13. Adaptively fitting the Venus model. (a): input model (b)-(d): results at different stages of fitting. (e) and (f): the final T-spline surface and the control mesh.

a visually satisfactory result (see Figure 13(d)). The final surface has 243 control points and is obtained in 10 iterations. Figure 13(e) and Figure 13(f) show the surface with the mapped pre-image and the T-mesh, respectively, and Table 2 provides the detailed information about the Venus model and the final periodic T-spline surface.

5 Conclusion

This paper has introduced a periodic B-spline basis function, with which periodic T-spline surfaces can be represented globally without the need of repeating the control points. This facilitates the surface fitting process where the control points are the unknowns. An algorithm of constructing periodic T-spline surfaces to approximate tubular triangular meshes has been presented. The use of T-splines promises an adaptive approach in that local refinement is carried out a the problematic regions. The choice of periodic splines, the special setting of the initial T-mesh and the use of the T-spline local refinement algorithm make the fitting process simple and efficient.

Meanwhile we also notice that there are still many venues for further improvement of the fitting algorithm. For example, how to incorporate geometric feature to guide fitting and how fine-tune some heuristic choices in the current algorithm are interesting questions, warranting further investigation.

Acknowledgments. This work is supported by the ARC 9/09 Grant (MOE2008-T2-1-075) of Singapore.

References

1. Dietz, U., Hoschek, J.: Smooth B-spline surface approximation to scattered. In: Hoschek, J., Dankwort, W. (eds.) Reverse Engineering, pp. 143–151. B.G. Teubner, Stuttgart (1996)
2. Eck, M., Hoppe, H.: Automatic reconstruction of B-spline surfaces of arbitrary topological type. In: Proceedings of SIGGRAPH 1996, pp. 325–334. ACM Press, New York (1996)
3. Krishnamurthy, V., Levoy, M.: Fitting smooth surfaces to dense polygon meshes. In: Proceedings of SIGGRAPH 1996, pp. 313–324. ACM Press, New York (1996)
4. Forsey, D.R., Bartels, R.H.: Surface fitting with hierarchical splines. ACM Transactions on Graphics 14, 134–161 (1995)
5. Greiner, G., Hormann, K.: Interpolating and approximating scattered 3D-data with hierarchical tensor product B-splines. In: Le Méhauté, A., Rabut, C., Schumaker, L.L. (eds.) Proceedings of Chamonix 1996, Vanderbilt University Press, Nashville (1997)
6. Huysmans, T., Sijbers, J., Versonk, B.: Parameterization of tubular surfaces on the cylinder. The Journal of WSCG 13, 97–104 (2005)
7. Wang, Y., Zheng, J.: Tubular triangular mesh parameterization and applications. Computer Animation and Virtual Worlds 21, 91–102 (2010)
8. Sederberg, T., Zheng, J., Bakenov, A., Nasri, A.: T-splines and T-NURCCs. ACM Transactions on Graphics (SIGGRAPH 2003) 22, 477–484 (2003)

9. Sederberg, T., Cardon, D., Finnigan, G., North, N., Zheng, J., Lyche, T.: T-spline simplification and local refinement. ACM Transactions on Graphics (SIGGRAPH 2004) 23, 276–283 (2004)
10. Zheng, J., Wang, Y., Seah, H.S.: Adaptive T-spline surface fitting to z-map models. In: Proceedings of GRAPHITE, Dunedin, New Zealand, pp. 405–411 (2005)
11. Li, W., Ray, N., Lévy, B.: Automatic and interactive mesh to T-spline conversion. In: Proceedings of the 4th Eurographics Symposium on Geometry Processing, pp. 191–200. Eurographics Association (2006)
12. Farin, G.E., Hansford, D.: The Essentails of CAGD. A K Peters, Ltd., Wellesley (2000)

Author Index