

Isabelle Debled-Rennesson
Eric Domenjoud
Bertrand Kerautret
Philippe Even (Eds.)

LNCS 6607

Discrete Geometry for Computer Imagery

16th IAPR International Conference, DGCI 2011
Nancy, France, April 2011
Proceedings



 Springer

The Springer logo, which consists of a white chess knight piece on a pedestal, followed by the word "Springer" in a white, serif font.

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Alfred Kobsa

University of California, Irvine, CA, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

TU Dortmund University, Germany

Madhu Sudan

Microsoft Research, Cambridge, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max Planck Institute for Informatics, Saarbruecken, Germany

Isabelle Debled-Rennesson Eric Domenjoud
Bertrand Kerautret Philippe Even (Eds.)

Discrete Geometry for Computer Imagery

16th IAPR International Conference, DGCI 2011
Nancy, France, April 6-8, 2011
Proceedings

Volume Editors

Isabelle Debled-Rennesson
Eric Domenjoud
Bertrand Kerautret
Philippe Even

LORIA, Equipe ADAGIo, Campus Scientifique
BP 239, 54506, Vandœuvre-lès-Nancy Cedex, France
E-mail: {isabelle.debled-rennesson, eric.domenjoud,
bertrand.kerautret, philippe.even}@loria.fr

ISSN 0302-9743

e-ISSN 1611-3349

ISBN 978-3-642-19866-3

e-ISBN 978-3-642-19867-0

DOI 10.1007/978-3-642-19867-0

Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011923068

CR Subject Classification (1998): I.3.5, I.4.1, I.4, I.3, G.2

LNCS Sublibrary: SL 6 – Image Processing, Computer Vision, Pattern Recognition,
and Graphics

© Springer-Verlag Berlin Heidelberg 2011

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

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 16th edition of the International Conference on Discrete Geometry for Computer Imagery was held in Nancy, France, April 6–8, 2011, and was organized by the ADAGIo team of the LORIA laboratory (Lorraine research center in computer science and its applications). DGCI 2011 attracted many researchers from all around the world. Indeed, 70 papers were submitted, from 22 different countries, confirming the international status of the conference. The contributions focus on models for discrete geometry, discrete and combinatorial topology, geometric transforms, discrete shape representation and analysis, discrete tomography and discrete and combinatorial tools for image segmentation and analysis.

Following a thorough reviewing process, remodeled for DGCI 2008, 40 papers were accepted and published in the present LNCS volume. Altogether, 20 papers were scheduled for oral presentation in single-track sessions, and 20 papers were presented as posters, with preliminary plenary sessions with very short oral presentations of these posters.

In addition, the program was enriched by three invited lectures, presented by internationally well-known speakers: Agnes Desolneux (CNRS researcher, MAP5, Paris Descartes University, France), Jarek Rossignac (Professor, GVV Center, Georgia Tech, USA), and Jean Serra (Professor Emeritus, ESIEE-LIGM, Paris, France).

For the first time, the 16th edition of DGCI hosted a demonstration session. The purpose of this session was to provide the opportunity to present and share effective applications related to the main topics of DGCI.

DGCI 2011 was supported by the International Association of Pattern Recognition (IAPR), and is the main conference associated with the Technical Committee on Discrete Geometry (TC18) of IAPR.

Hereby, we would like to thank all contributors who responded to the call for papers of DGCI, the invited speakers, all reviewers and members of the Steering and Program Committees, as well as all participants of DGCI. We would also like to express our gratitude to the LORIA Laboratory and INRIA Nancy-Grand Est (French National Institute for Research in Computer Science and Control) for hosting this event and providing all the necessary facilities. We finally thank our sponsoring institutions for providing the financial support essential for a successful event.

April 2011

Isabelle Debled-Rennesson
Eric Domenjoud
Philippe Even
Bertrand Kerautret

Organization

Organizing Committee

Isabelle Debled-Rennesson	LORIA, University of Nancy, (General Chair)
Eric Domenjoud	LORIA, CNRS, (Co-chair)
Philippe Even	LORIA, University of Nancy
Bertrand Kerautret	LORIA, University of Nancy, (Co-chair and Webmaster)
Anne-Lise Charbonnier	LORIA, INRIA, (Local Arrangements)

Steering Committee

Eric Andres	XLIM-SIC, University of Poitiers, France
Gunilla Borgefors	Centre for Image Analysis, University of Uppsala, Sweden
Srečko Brlek	LaCIM, UQAM, Montréal (QC), Canada
Jean-Marc Chassery	Gipsa-Lab, CNRS, Grenoble, France
David Coeurjolly	LIRIS, CNRS, Lyon, France
Ullrich Köthe	HCI, University of Heidelberg, Germany
Annick Montanvert (President)	Gipsa-Lab, University of Grenoble, France
Kálmán Palágyi	Department of Image Processing and Computer Graphics, University of Szeged, Hungary
Gabriella Sanniti di Baja	Istituto di Cibernetica Eduardo Caianiello, CNR, Naples, Italy

Program Committee

Renata Barneva	Department of Computer and Information Sciences, SUNY Fredonia (NY), USA
Achille Braquelaire	LaBRI, University of Bordeaux, France
Michel Couprie	Computer Science Department, ESIEE, Paris, France
Christophe Fiorio	LIRMM, CNRS-University of Montpellier, France
Leila De Floriani	DISI, University of Genova, Italy
Atsushi Imiya	Institute of Media and Information Technology, University of Chiba, Japan
Pieter Jonker	TU Delft, The Netherlands
Christer Kiselman	Department of Mathematics, University of Uppsala, Sweden
Walter Kropatsch	Institute of Computer Graphics and Algorithms, TU Vienna, Austria

Jacques-Olivier Lachaud	LAMA, University of Savoie, Chambéry, France
Rémy Malgouyres	LIMOS, University of Auvergne, Clermont-Ferrand, France
Renzo Pinzani	Dipartimento di Sistemi e Informatica, Firenze, Italy
Robin Strand	Centre for Image Analysis, University of Uppsala, Sweden
Edouard Thiel	LIF, University of Aix-Marseille, France
Peter Veelaert	University College of Ghent, Belgium

Referees

Eric Andres	Eric Domenjoud	Thierry Monteil
Dominique Attali	Ulrich Eckhardt	Benoît Naegel
Elena Barucci	Philippe Even	Thanh Phuong Nguyen
Reneta Barneva	Thomas Fernique	Nicolas Normand
K. Joost Batenburg	Fabien Feschet	László Nyúl
Nicolas Bédaride	Christophe Fiorio	Kálmán Palágyi
Valérie Berthé	Laurent Fuchs	Nicolas Passat
Gilles Bertrand	Yan Gérard	Samuel Peltier
Isabelle Bloch	Luc Gillibert	Renzo Pinzani
Olivier Bodini	Jean-Pierre Guédon	Stéphanie Prévost
Gunilla Borgefors	Yll Haxhimusa	Xavier Provençal
Achille Braquelaire	Atsushi Imiya	Laurent Provot
Valentin Brimkov	Yukiko Kenmochi	Eric Rémy
Srećko Brlek	Bertrand Kerautret	Jean-Pierre Reveillès
Alfred Bruckstein	Nahum Kiryati	Christian Ronse
Luc Brun	Christer Kiselman	Tristan Roussillon
Sara Brunetti	Reinhard Klette	Gabriella Sanniti di Baja
Jean-Marc Chassery	Yung Kong	Isabelle Sivignon
David Coeurjolly	Ullrich Köthe	Pierre Soille
Michel Couprie	Walter Kropatsch	Robin Strand
Jean Cousty	Jacques-Olivier Lachaud	Stina Svensson
José Crespo	Gaëlle Largeteau-Skapin	Mohamed Tajine
Marie-Andrée	Bruno Lévy	Hugues Talbot
Jacob-Da Col	Pascal Lienhardt	Alexandru Telea
Guillaume Damiand	Joakim Lindblad	Edouard Thiel
Leila De Floriani	Laurent Lucas	Laure Tougne
François De Vieilleville	Grégoire Malandain	Jean-Luc Toutant
Isabelle	Rémy Malgouyres	Sébastien Valette
Debled-Rennesson	Robert Melter	Peter Veelaert
Pascal Desbarats	Christian Mercat	Anne Vialard
Olivier Devillers	Serge Miguet	Laurent Vuillon
Jean-Philippe Domenger	Annick Montanvert	Laurent Wendling

Sponsoring Institutions

DGCI 2011 was supported by the following institutions:

- The International Association for Pattern Recognition (IAPR)
- GdR Informatique Mathématique (GdR IM)
- Université Henri Poincaré, Nancy 1 (UHP)
- Université Nancy 2
- Institut National Polytechnique de Lorraine (INPL)
- Ministère de l'Enseignement Supérieur et de la Recherche (MESR)
- La région Lorraine
- La Communauté Urbaine du Grand Nancy (CUGN)

Table of Contents

Invited Speakers

A Probabilistic Grouping Principle to Go from Pixels to Visual Structures	1
<i>Agnès Desolneux</i>	
Ball-Based Shape Processing	13
<i>Jarek Rossignac</i>	
Hierarchies and Optima	35
<i>Jean Serra</i>	

Models for Discrete Geometry

An Arithmetic and Combinatorial Approach to Three-Dimensional Discrete Lines	47
<i>Valérie Berthé and Sébastien Labbé</i>	
Smooth 2D Coordinate Systems on Discrete Surfaces	59
<i>Colin Cartade, Rémy Malgouyres, Christian Mercat, and Chafik Samir</i>	
An Improved Riemannian Metric Approximation for Graph Cuts	71
<i>Ondřej Daněš and Pavel Matula</i>	
Introduction to Digital Level Layers	83
<i>Yan Gérard, Laurent Provot, and Fabien Feschet</i>	
Another Definition for Digital Tangents	95
<i>Thierry Monteil</i>	
Associating Cell Complexes to Four Dimensional Digital Objects	104
<i>Ana Pacheco and Pedro Real</i>	
Metric Bases for Polyhedral Gauges	116
<i>Fabien Rebatel and Édouard Thiel</i>	

Discrete and Combinatorial Topology

Completions and Simplicial Complexes	129
<i>Gilles Bertrand</i>	
Hierarchic Euclidean Skeletons in Cubical Complexes	141
<i>Michel Couprie</i>	

Well-Composed Cell Complexes 153
Rocio Gonzalez-Diaz, Maria-Jose Jimenez, and Belen Medrano

A Unified Topological Framework for Digital Imaging 163
Loïc Mazo, Nicolas Passat, Michel Couprie, and Christian Ronse

Isthmus-Based 6-Directional Parallel Thinning Algorithms 175
Benjamin Raynal and Michel Couprie

Geometric Transforms

Quasi-Linear Transformations, Numeration Systems and Fractals 187
Marie-Andrée Jacob-Da Col and Pierre Tellier

Path-Based Distance with Varying Weights and Neighborhood Sequences 199
Nicolas Normand, Robin Strand, Pierre Evenou, and Aurore Arlicot

Sparse Object Representations by Digital Distance Functions 211
Robin Strand

Discrete Shape Representation, Recognition and Analysis

Efficient Robust Digital Hyperplane Fitting with Bounded Error 223
Dror Aiger, Yukiko Kenmochi, Hugues Talbot, and Lilian Buzer

Analytical Description of Digital Circles 235
Eric Andres and Tristan Roussillon

Circular Arc Reconstruction of Digital Contours with Chosen Hausdorff Error 247
Bertrand Kerautret, Jacques-Olivier Lachaud, and Thanh Phuong Nguyen

Recursive Calculation of Relative Convex Hulls 260
Gisela Klette

An Error Bounded Tangent Estimator for Digitized Elliptic Curves 272
Dilip K. Prasad, Raj Kumar Gupta, and Maylor K.H. Leung

Estimation of the Derivatives of a Digital Function with a Convergent Bounded Error 284
Laurent Provot and Yan Gérard

Properties and Applications of the Simplified Generalized Perpendicular Bisector 296
Aurélie Richard, Gaëlle Largeteau-Skapin, Marc Rodríguez, Eric Andres, Laurent Fuchs, and Jean-Serge Dimitri Ouattara

Delaunay Properties of Digital Straight Segments	308
<i>Tristan Roussillon and Jacques-Olivier Lachaud</i>	
Computing the Characteristics of a SubSegment of a Digital Straight Line in Logarithmic Time	320
<i>Mouhammad Said and Jacques-Olivier Lachaud</i>	
A Near-Linear Time Guaranteed Algorithm for Digital Curve Simplification under the Fréchet Distance	333
<i>Isabelle Sivignon</i>	
Distance between Separating Circles and Points	346
<i>Peter Veelaert</i>	
Optimal Consensus Set for Annulus Fitting	358
<i>Rita Zrour, Gaëlle Largeteau-Skapin, and Eric Andres</i>	

Discrete Tomography

Bounds on the Difference between Reconstructions in Binary Tomography	369
<i>K. Joost Batenburg, Wagner Fortes, Lajos Hajdu, and Robert Tijdeman</i>	
Tiling the Plane with Permutations	381
<i>Alexandre Blondin Massé, Andrea Frosini, Simone Rinaldi, and Laurent Vuillon</i>	
Characterization of $\{-1,0,+1\}$ Valued Functions in Discrete Tomography under Sets of Four Directions	394
<i>Sara Brunetti, Paolo Dulio, and Carla Peri</i>	
Growth of Discrete Projection Ghosts Created by Iteration	406
<i>Imants Svalbe and Shekhar Chandra</i>	
Properties of Minimal Ghosts	417
<i>Imants Svalbe and Nicolas Normand</i>	

Morphological Analysis

Mathematical Morphology on Hypergraphs: Preliminary Definitions and Results	429
<i>Isabelle Bloch and Alain Bretto</i>	
Some Morphological Operators on Simplicial Complex Spaces	441
<i>Fábio Dias, Jean Cousty, and Laurent Najman</i>	
Selection of Relevant Nodes from Component-Trees in Linear Time	453
<i>Nicolas Passat and Benoît Naegel</i>	

Discrete and Combinatorial Tools for Image Segmentation and Analysis

Image Denoising with a Constrained Discrete Total Variation Scale Space.....	465
<i>Igor Ciril and Jérôme Darbon</i>	
Smale-Like Decomposition and Forman Theory for Discrete Scalar Fields	477
<i>Lidija Čomić, Mohammed Mostefa Mesmoudi, and Leila De Florianì</i>	
ACCORD: With Approximate Covering of Convex Orthogonal Decomposition.....	489
<i>Mousumi Dutt, Arindam Biswas, and Partha Bhowmick</i>	
Measures for Surface Comparison on Unstructured Grids with Different Density	501
<i>Natalia Dyshkant</i>	
Approximate Shortest Paths in Simple Polyhedra	513
<i>Fajie Li and Reinhard Klette</i>	
Author Index	525

A Probabilistic Grouping Principle to Go from Pixels to Visual Structures

Agnès Desolneux

MAP5 (UMR CNRS 8145), University Paris Descartes,
45 rue des Saints-Pères, 75006 Paris, France

Abstract. We will describe here how the Helmholtz principle, which is a principle of visual perception, can be translated into a computational tool that can be used for many problems of discrete image analysis. The Helmholtz principle can be formulated as “we immediately perceive whatever has a low likelihood of resulting from accidental arrangement”. To translate this principle into a computational tool, we will introduce a variable called NFA (Number of False Alarms) associated to any geometric event in an image. The NFA of an event is defined as the expectation of the number of occurrences of this event in a pure noise image of same size. Meaningful events will then be events with a very low NFA. We will see how this notion can be efficiently used in many detection problems (alignments, smooth curves, edges, etc.). The common framework of these detection problems is that they can all be translated into the question of knowing whether a given group of pixels is meaningful or not. This is a joint work with Lionel Moisan and Jean-Michel Morel.

Keywords: grouping laws, Gestalt theory, Helmholtz principle, rare events, alignments, edge detection, segmentation.

1 Introduction

When one looks at an image, one usually can see in it many geometric structures (straight segments, curves, homogeneous regions, etc.). But these objects are not really present in the image, they are only the result of our visual perception that is able to group pixels together according to some geometric criteria. Now, how can this grouping phenomenon be translated into a mathematical and computational principle in order to make a computer “see” geometric structures in an image? This can be achieved by formalizing and using the so-called *Helmholtz principle*, that we explain now.

1.1 Helmholtz Principle

The Helmholtz principle is a principle of visual perception that can be formulated two ways:

1. The first way is common sensical. It simply states that “*we do not perceive any structure in a uniform random image*”. In this form, the principle was first stated by Attneave in 1954 [1].

2. In its stronger form, the Helmholtz principle states that whenever some large deviation from randomness occurs, a structure is perceived. In other words: “*we immediately perceive whatever has a low likelihood of resulting from accidental arrangement*”. It has been first stated under this form in Computer Vision by S.-C. Zhu [2] and D. Lowe [3].

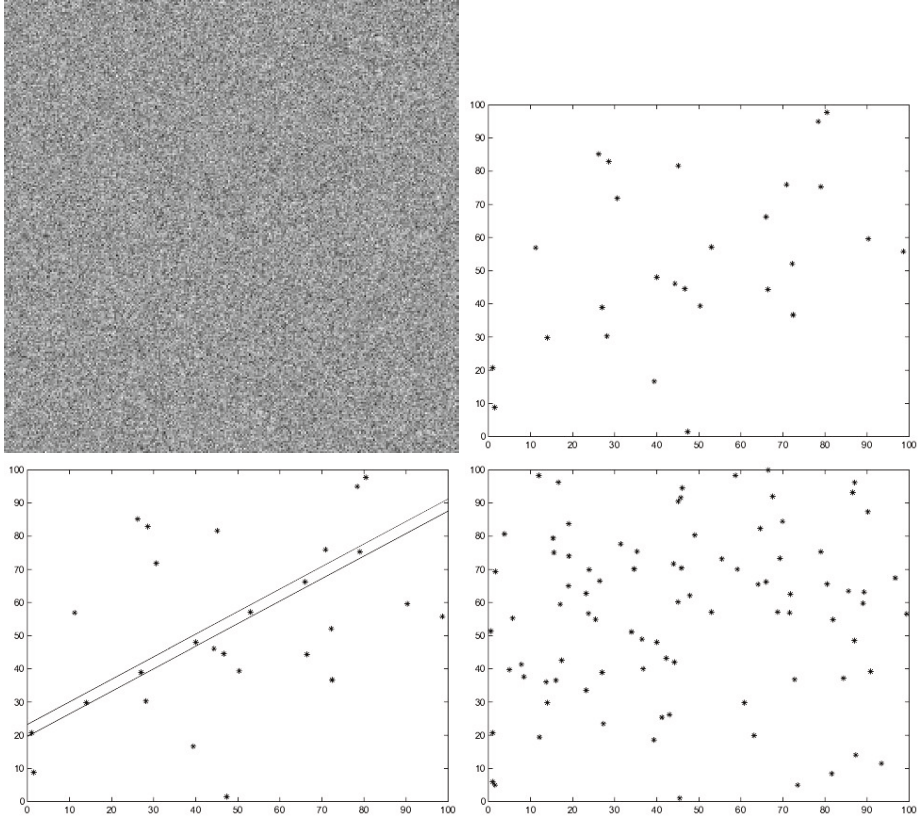


Fig. 1. Top left: an image of pure noise (pixels have independent identically distributed grey levels following the uniform distribution on $\{0, 1, \dots, 255\}$). In this image, no visual structure is perceived. Top right: an image made of 27 points, such that 7 of them are aligned and the 20 others are randomly positioned. The alignment is immediately perceived and will be detected by the *a contrario* method we will present (result on the bottom left figure). Bottom right: when there are 80 random points instead of 20, the fact to have 7 points aligned is not a rare event anymore, and we don't perceive it, even if it is there.

Given a geometric structure, Helmholtz principle tells us that we immediately perceive it if it has a low probability of resulting from randomness. But what

are the “interesting” structures for our visual perception? This question (among others) has been studied by the Gestalt School of Psychophysiology. We briefly recall part of their work in the following section.

1.2 Gestalt Theory

Before the development of Gestalt Theory, there were many psychophysiological experiments based on optic-geometric illusions (see for instance the left part of Figure 2). The aim of these illusions is to ask: “what is the reliability of our visual perception?” But Gestalt theory (developed by Wertheimer, Metzger, Kanizsa - see [4] for instance) does not continue on the same line. The question is not why we sometimes see a distorted line when it is straight; the question is why we do see a line at all. This perceived line is the result of a construction process. Gestalt theory starts with the assumption that there is a small list of active grouping laws in visual perception: vicinity, same attribute (like colour, shape, size or orientation), alignment, good continuation, symmetry, parallelism, convexity, closure, constant width, amodal completion, T-junctions, X-junctions, Y-junctions. Moreover, all grouping Gestalt laws are *recursive*: they can be applied first to atomic inputs and then in the same way to partial Gestalts already constituted. This is illustrated by the right part of Figure 2.

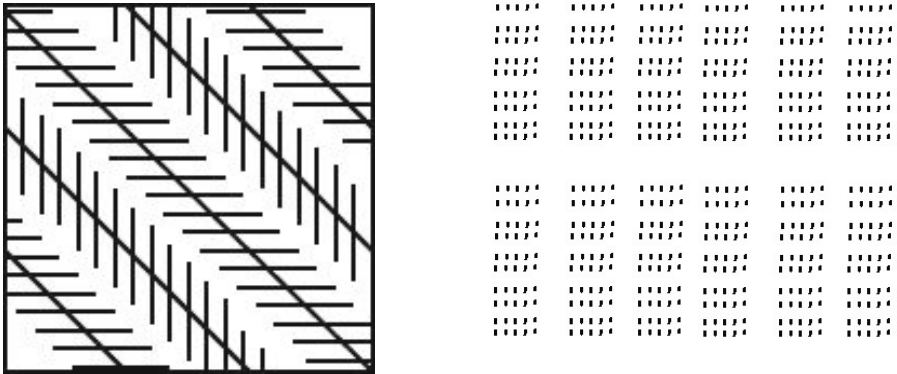


Fig. 2. Left: Zoellner’s Illusion (1860). Right: the same Gestalt grouping laws namely alignment, parallelism, constant width and proximity, are recursively applied not less than six times.

2 A Computational Tool: The Number of False Alarms

Having now a grouping principle (Helmholtz principle) and visually relevant structures in images (the ones that obey Gestalt grouping laws), we can combine this into a general framework which is the one of the so-called *a contrario*

methodology. The general description of this methodology is the following. Given n geometric objects O_1, \dots, O_n , let X_i be a variable describing an attribute (position, colour, orientation, size, etc...) of O_i . Then define a Null Hypothesis \mathcal{H}_0 (that is a noise model also called a *contrario* model): X_1, \dots, X_n are independant identically distributed. Now, consider an observed geometric event E concerning k of the objects (for instance if X_i are spatial positions we observe X_1, \dots, X_k are very close). The question is: can this observed geometric event E happen by chance? In other words, what is its likelihood under \mathcal{H}_0 ? To answer this, we define a computational tool called *Number of False Alarms* that is defined as the expected number of occurrences of the event E under \mathcal{H}_0 :

$$\text{NFA}(E) := \mathbb{E}_{\mathcal{H}_0}[\text{number of occurrences of the observed event}].$$

If the statistical test “ $\text{NFA}(E) \leq \varepsilon$ ” (for ε fixed and small, meaning $\varepsilon \leq 1$) is positive then E is said to be an ε -*meaningful event*. When $\varepsilon = 1$, we simply talk about *meaningful event*.

We will see in the following through many different exemples that the NFA defined above is a universal variable adaptable to many detection problems. A detailed presentation of the whole *a contrario* methodology and its applications can be found in the book [5].

3 Examples

3.1 Alignments in an Image

The first example of the *a contrario* method we give here is the detection of alignments in a discrete image [6]. It corresponds to the grouping of pixels according to the parallelism of their orientation. Let us detail this.

Given a discrete image of size $N \times N$ pixels, at each pixel, we can compute an orientation (\perp to the gradient). The noise model \mathcal{H}_0 is defined by: pixels at distance ≥ 2 have i.i.d. orientations, uniformly distributed on $[0, 2\pi)$.

Definition 1 (Meaningful segment). *Let S be a sequence of l consecutive and aligned pixels such that k of them have their orientation aligned with the one of the segment at a given precision p (see Figure 3). We say that the segment S is ε -meaningful if:*

$$\text{NFA}(S) = N^4 \times \mathcal{B}(l, k, p) = N^4 \sum_{j=k}^l \binom{l}{j} p^j (1-p)^{l-j} \leq \varepsilon.$$

Let us explain the formula for the NFA. The first term is the number of tests that we make: it is thus the total number of discrete straight segment in the image, that is $\simeq N^4$. The second term is the binomial tail: it is the probability that, under the noise model, at least k pixels among l pixels have their orientation aligned with the orientation of the segment according to the precision p .

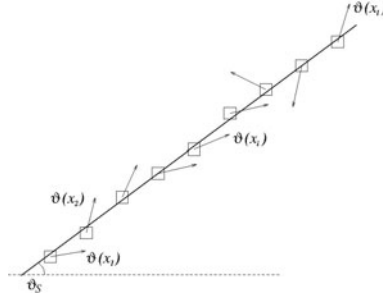


Fig. 3. A discrete straight segment made of l pixels taken at distance 2. At each pixel x_i , there is an orientation $\theta(x_i)$ and it is said to be aligned with the orientation θ_S of the segment according to precision p when $|\theta(x_i) - \theta_S| \leq p\pi$.

The main property is then that the expected number of ε -meaningful segments in a pure noise image of size $N \times N$ pixels is less than ε .

Now, when a segment is very meaningful (meaning that $\text{NFA}(S) \ll \varepsilon$), then many segments it contains, or is contained in, are also meaningful. We thus need a kind of “selection rule”, or “minimal representation rule” to keep only the “best representatives”. Thanks to the NFA, we can compare segments, and we then decide to look only at segments which are local minima of the NFA, in the sense of the following definition.

Definition 2 (Maximal Meaningful segment). *A segment S is maximal meaningful if it is meaningful and if $\forall S' \subset S$ (resp. $S' \supset S$), then $\text{NFA}(S') \geq \text{NFA}(S)$ (resp. $\text{NFA}(S') > \text{NFA}(S)$).*

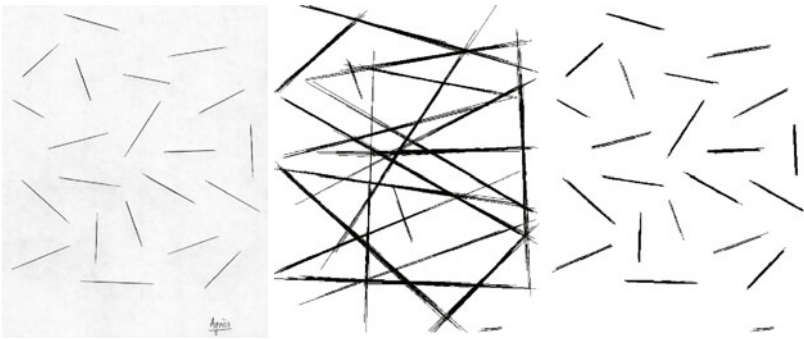


Fig. 4. From left to right: the original image, all meaningful segments, maximal meaningful segments

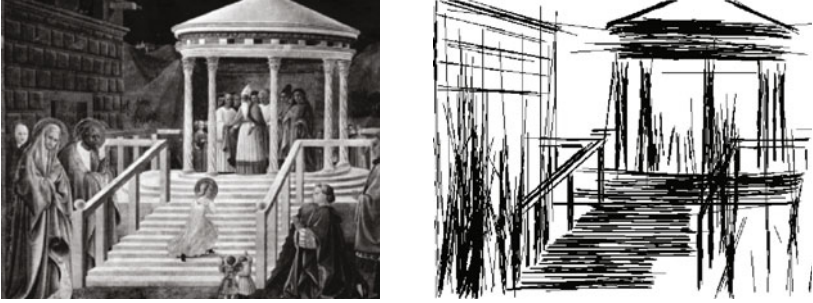


Fig. 5. An image (scan from a painting by Uccello), and its maximal meaningful alignments

3.2 Meaningful Boundaries

A second application of the *a contrario* method is the very general and very studied problem of “edge detection” in an image. The aim is to divide the image into “homogeneous” regions (this is the problem of Image Segmentation in Computer Vision). And the dual approach consists in finding the boundaries of these regions - as “highly” contrasted curves (this is the problem of edge detection).

As for the meaningful alignment, we first need to define the structures we look for and also the noise model. At each pixel x of an image u of size $N \times N$, we start by defining the contrast $c(x)$ by $c(x) = |\nabla u|(x)$. And then, the noise model is simply given by the empirical distribution of the contrast in the image with the additional independance hypothesis, which means that the probability that a pixel has a contrast larger than μ is given by

$$H(\mu) = \mathbb{P}(c(x) \geq \mu) = \frac{1}{N^2} \#\{y / |\nabla u|(y) \geq \mu\},$$

and pixels at distance larger than 2 are assumed (in the noise model) to have independant contrasts.

Finally, what are the candidate to be edge curves in the image ? It is not possible to look at all the curves in the image, and good candidates are the level lines of the image (defined as the boundaries of the level sets). Let thus N_l be the number of level lines in the image. We then have the following definition.

Definition 3 (Meaningful boundaries). *Let \mathcal{C} be a level line of the image, with length l and minimal contrast μ . We say that \mathcal{C} is an ε -meaningful boundary if*

$$\text{NFA}(\mathcal{C}) = N_l \times H(\mu)^l \leq \varepsilon.$$

For the same reasons as the ones explained for meaningful alignments, we also need here a notion of maximality. We can then define maximal meaningful boundaries as local minima of the NFA for the relation of inclusion of level sets (see details in [7]). Also the same method can be applied to pieces of level lines (instead of whole level lines), and we then obtain *meaningful edges*.

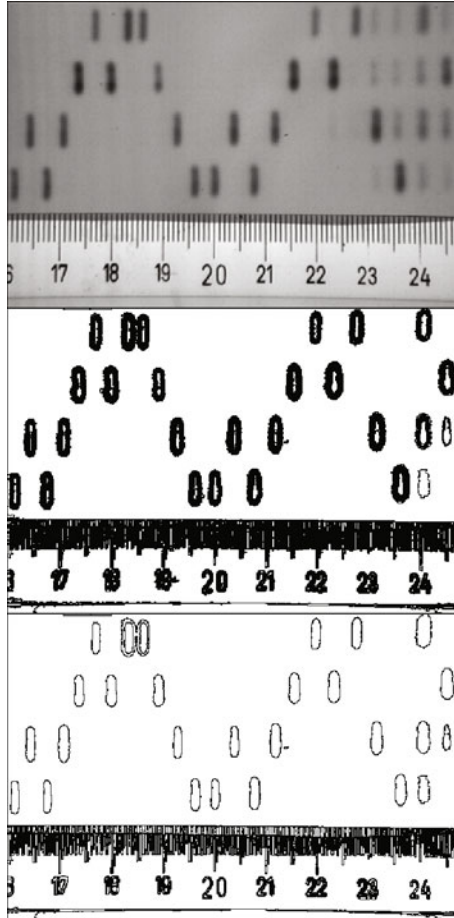


Fig. 6. Top: original image. Middle: all meaningful boundaries. Bottom: maximal meaningful boundaries.

3.3 Meaningful Good Continuations

Instead of looking at the contrast across a level line, one can look at its regularity and see if it is more regular than “what we would expect in pure noise”. This has been formalized by F. Cao in [8], and to keep the Gestalt theory term, such curves are called “good continuations”. Thus, the aim is to look for meaningful “smooth” curves, without any contrast information.

Let $\Gamma = (p_0, \dots, p_{l+1})$ be a discrete curve of length l , and let θ be its maximal discrete curvature (see also Figure 7):

$$\theta = \max_{1 \leq i \leq l} |\text{angle}(p_{i+1} - p_i, p_i - p_{i-1})|.$$

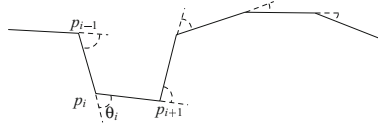


Fig. 7. A discrete curve is a sequence \$(p_0, \dots, p_{l+1})\$ of points, and its maximal discrete curvature can be defined as \$\theta = \max_{1 \leq i \leq l} |\theta_i|\$

The noise model here is that the angles are i.i.d. uniform on \$[0, 2\pi)\$, *i.e.* the curve is a “random walk”. Let \$N_c\$ be the number of considered curves (usually the number of pieces of level lines in the image).

Definition 4 (meaningful good-continuation). We say that a discrete curve \$\Gamma\$ is an \$\varepsilon\$-meaningful good-continuation if

$$\theta < \frac{\pi}{2} \quad \text{and} \quad \text{NFA}(\Gamma) = N_c \left(\frac{\theta}{\pi} \right)^l < \varepsilon.$$

Again, we have a definition of maximality: a meaningful good-continuation \$\Gamma\$ is maximal meaningful if: \$\forall \Gamma' \subset \Gamma\$, \$\text{NFA}(\Gamma') \geq \text{NFA}(\Gamma)\$ and \$\forall \Gamma' \supsetneq \Gamma\$, \$\text{NFA}(\Gamma') > \text{NFA}(\Gamma)\$. An interesting property is then that: if \$\Gamma\$ and \$\Gamma'\$ are two maximal meaningful good-continuations on the same level line, then \$\Gamma \cap \Gamma' = \emptyset\$.

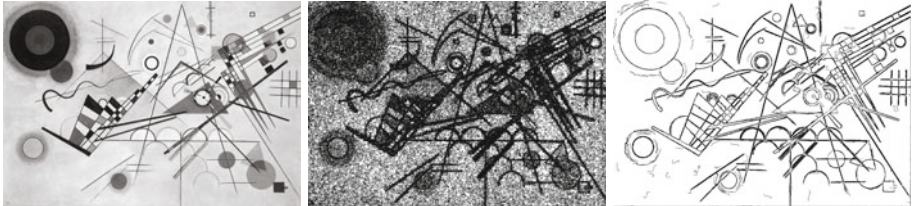


Fig. 8. From left to right: the original Image (painting by Kandinsky), all the level lines (with some quantization step for grey levels), maximal meaningful good-continuations

3.4 Similarity of a Scalar Attribute

Another application of the *a contrario* methodology is the grouping of objects according to any scalar attribute (like grey level, or orientation, etc.) More precisely, assume we have \$M\$ “objects”, and each of them as an attribute \$q \in \{1, 2, \dots, L\}\$. Let \$a \leq b\$ be two attribute values and let \$G\$ be the group of objects (among the \$M\$ objects) such that their scalar attribute \$q\$ satisfies \$a \leq q \leq b\$. Then denote \$k\$ the cardinality of \$G\$ and define its NFA under the noise model that attribute values are i.i.d. uniform, by

$$\text{NFA}(G) = \text{NFA}([a, b]) = \frac{L(L+1)}{2} \cdot \mathcal{B} \left(M, k, \frac{b-a+1}{L} \right).$$



Fig. 9. From left to right: image (INRIA) of the church of Valbonne, maximal meaningful good-continuations, maximal meaningful boundaries

(In this formula, notice that $L(L + 1)/2$ corresponds to the total number of possible groups that can be made this way, and \mathcal{B} denotes again, as in the case of meaningful alignments, the tail of the binomial distribution). The group G (or, in an equivalent way, the interval $[a, b]$) is said ε -meaningful when $\text{NFA}(G) \leq \varepsilon$. And again, using the relation of inclusion of intervals, we can define maximal meaningful groups (intervals).

A first application is the study of an image grey level histogram. Indeed, looking for the maximal meaningful intervals of this histogram is a way to obtain an automatic grey level quantization of the image. See an illustration of this on Figure 10.

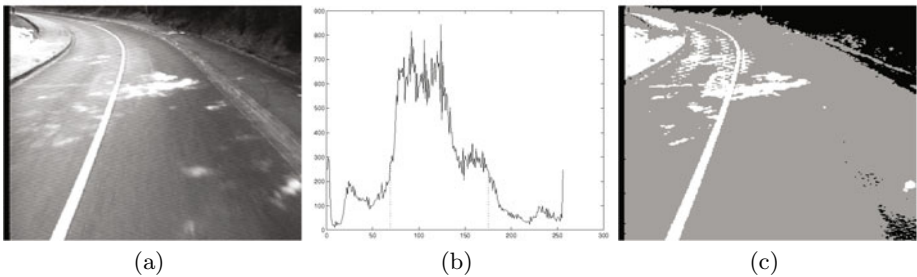


Fig. 10. Maximal meaningful intervals and optimal grey level quantization of a digital image. (a) original image. (b) histogram of grey-levels of the image with its single maximal meaningful interval $[69, 175]$ (between the dotted lines). (c) Quantized image: black points represent points in the original image with grey level in $[0, 68]$, grey points represent points with grey level in the maximal meaningful interval $[69, 175]$ and white points represent points with grey-level larger than 176.



Fig. 11. Uccello’s painting: maximal meaningful alignments and histogram of orientations. Two maximal meaningful intervals are found in this histogram corresponding respectively to the horizontal and vertical segments.

Instead of using a uniform assumption for the noise model distribution of attributes, we can more generally use any distribution or class of distributions. For instance, we can define meaningful groups according to an attribute that is assumed to have a decreasing distribution (like the length, the area, etc.) by setting for the Number of False Alarm of an interval $[a, b]$:

$$\text{NFA}([a, b]) = \frac{L(L + 1)}{2} \cdot \max_{p \in \mathcal{D}} \mathcal{B} \left(M, k, \sum_{i=a}^b p(i) \right)$$

where k is the number of objects having their attribute value in $[a, b]$ (i.e. it is the cardinality of G) and \mathcal{D} is the set of decreasing distributions on $\{1, \dots, L\}$.

An example of application is the recursivity of grouping as formulated by the Gestalt theory. See Figures [11](#) and [12](#) for some illustrations and comments of this.

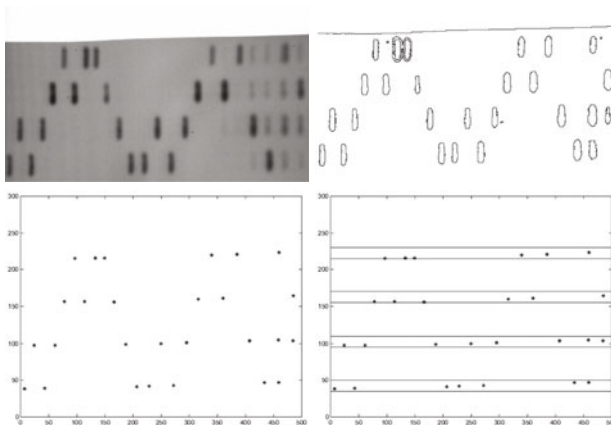


Fig. 12. Gestalt grouping principles at work for building an “order 3” Gestaltist description of the image (alignment of blobs of the same size). First row: original DNA image (left) and its maximal meaningful boundaries (right). Second row: left, barycenters of all meaningful regions whose area is inside the only maximal meaningful interval of the histogram of areas; right, meaningful alignments of these points.

4 Conclusion

Through many illustrations, we have seen that Helmholtz principle combined with Gestalt grouping laws can become, through the definition of a Number of False Alarms, a powerful computational tool. It can then be used in many applications in detection problems, but also in shape recognition (Musé et al. [9]); image matching (Rabin et al. [10]); epipolar geometry (Moisan and Stival [11]), motion detection and analysis (Veit et al. [12]); clustering (Cao et al. [13]); stereovision (Sabater et al. [14]); image denoising (by grain filters, Coupier et al. [15]); etc. The main advantage of the whole *a contrario* methodology is that, thanks to the framework of statistical testing, it provides a validation of found structures and also a way to set the different thresholds of a given problem in an automatic way. An interesting research direction is then the comparison of these predicted thresholds with the ones of our visual perception. Also from a mathematical point of view, the *a contrario* methodology raises many difficult questions of stochastic geometry.

Now, even if the *a contrario* method is very general and has many applications, some open questions still remain. A first question is: how to deal with the “over-determination” of images (i.e. the fact that visual objects usually have several qualities at the same time)? This would require the definition of a generalized computational tool like the NFA that would be able to deal with several grouping criteria at the same time. A second open question is: how to deal with “conflicts” of qualities? Since there is no “universal hierarchy” of qualities, we cannot hope for any reliable explanation of a figure by summing up the results of one or several partial gestalts detector. Only a global synthesis, treating all conflicts of partial gestalts, can give the correct result. This would require the need of another framework than Helmholtz principle (like for instance the Minimum Description Length principle or Bayesian compositional approaches [16], [17]).

References

1. Attneave, F.: Some informational aspects of visual perception. *Psychological Review* 61, 183–193 (1954)
2. Zhu, S.C.: Embedding Gestalt Laws in Markov Random Fields. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 21(11), 1170–1187 (1999)
3. Lowe, D.: *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, Dordrecht (1985)
4. Kanizsa, G.: *Grammatica del Vedere / La Grammaire du Voir*. Il Mulino, Bologna / Éditions Diderot, arts et sciences (1980/1997)
5. Desolneux, A., Moisan, L., Morel, J.-M.: *From Gestalt Theory to Image Analysis: A Probabilistic Approach*. Springer, Heidelberg (2008)
6. Desolneux, A., Moisan, L., Morel, J.-M.: Meaningful Alignments. *Int. Journal of Computer Vision* 40(1), 7–23 (2000)
7. Desolneux, A., Moisan, L., Morel, J.-M.: Edge Detection by Helmholtz Principle. *Journal of Mathematical Imaging and Vision* 14(3), 271–284 (2001)
8. Cao, F.: Good continuation in digital images. In: *Int. Conf. Computer Vision (ICCV)*, vol. 1, pp. 440–447 (2003)

9. Musé, P., Sur, F., Cao, F., Gousseau, Y.: Unsupervised thresholds for shape matching. In: *Int. Conf. on Image Processing (ICIP 2003)*, vol. 2, pp. 647–650 (2003)
10. Rabin, J., Delon, J., Gousseau, Y.: A statistical approach to the matching of local features. *SIAM Journal on Imaging Sciences* 2(3), 931–958 (2009)
11. Moisan, L., Stival, B.: A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix. *International Journal of Computer Vision* 57(3), 201–218 (2004)
12. Veit, T., Cao, F., Bouthemy, P.: An a contrario decision framework for region-based motion detection. *International Journal of Computer Vision* 68(2), 163–178 (2006)
13. Cao, F., Delon, J., Desolneux, A., Musé, P., Sur, F.: A unified framework for detecting groups and application to shape recognition. *Journal of Mathematical Imaging and Vision* 27(2), 91–119 (2007)
14. Sabater, N., Almansa, A., Morel, J.-M.: Rejecting wrong matches in stereovision. Technical report CMLA, ENS Cachan, No. 2008-28 (2008)
15. Coupier, D., Desolneux, A., Ycart, B.: Image denoising by statistical area thresholding. *Journal of Mathematical Imaging and Vision* 22(2-3), 183–197 (2005)
16. Bienenstock, E., Geman, S., Potter, D.: Compositionality, MDL Priors, and Object Recognition. In: *Mozer, M.C., Jordan, M.I., Petsche, T. (eds.) Advances in Neural Information Processing Systems*, vol. 9, pp. 838–844. MIT Press, Cambridge (1997)
17. Zhu, S.C., Mumford, D.: A Stochastic Grammar of Images. *Foundations and Trends in Computer Graphics and Vision* 2(4), 259–362 (2006)

Ball-Based Shape Processing

Jarek Rossignac

School of Interactive Computing, Georgia Institute of Technology, Atlanta, USA
rossignac@gatech.edu

Abstract. Constant radius offsetting and blending operations are important for digital shape and image processing. They may be formulated using Minkowski sums with a ball of fixed radius. We review their extensions to variable distance offsetting. Specifically, we compare three different formulations of variable distance offsetting for planar shapes: orthogonal, radial, and ball. We discuss compatibility conditions that specify when a shape is the offset of another. We also discuss the applications of these formulations for computing the average and morph of two shapes and the centerline of an elongated shape. Finally, we discuss a set theoretic formulation of a variable radius blending of a shape.

Keywords: Variable Radius Offsetting. Rounding and filleting. Medial Axis. Skeleton extraction. Image Segmentation. Shape Averaging. Shape Morphing. Shape correspondence. Tangent Balls. Ball Map. Ball Morph.

1 Introduction

The offset of a shape in the plane plays a central role in shape processing. **Constant distance offsets** (also called constant radius offsets) have been used for modeling safety regions around a shape, for detecting interferences, for simulating growth, for measuring distances and discrepancies between shapes, for defining tolerances on parametric models, and for blending (i.e., removing) sharp corners and narrow passages in the shape or in its complement.

Specializing Serra and Matheron's [1,2] **opening** and **closing** to cases where the structuring element is a ball, Rossignac and Requicha have formulated these **constant-radius blending** operations [3,4] as combinations of **growing** (positive offsetting) and **shrinking** (negative offsetting). Depending on the order, such a combination can either blend all concave or all convex corners, but is not guaranteed to remove both types of sharp features. To address this lack of symmetry and to provide a solutions that blends both concave and convex corners, Williams and Rossignac have defined the **mortar** [5] of the shape as the fill (i.e., grow-shrink) of its boundary and the **tightening** [6,7] of a shape as the result of minimizing the length of the boundary while keeping it inside the mortar.

Variable distance offsets are convenient for expressing one shape as a variation (offset) of another. We discuss here three different formulations of such offsets: orthogonal, radial, and ball.

The **orthogonal offset** [8,9] displaces each point of the boundary of the shape by a prescribed distance in the normal direction.

The **radial offset** adjusts that direction based on the derivative of the distance function and produces a new boundary that is the subset of the envelop [10] of a family of disks with centers on the boundary of the shape and with radii equal to the offset distance value associated with the center point.

The **ball offset** (also called **tangent-ball offset**) recently introduced by Chazal, Lieutier, Rossignac, and Whited [11] produces a new boundary that is the subset of the envelop of a family of disks that are tangent to the original boundary and have as radius half the offset distance associated with the tangent point.

When the distance is constant, all three formulations yield the same result as the constant distance offset. However, when the offset distance varies along the boundary of the shape, each formulation produces a different result and has its advantages.

For example, positive and negative orthogonal offsets of a curve are sometimes used to formulate a variable width stroke in terms of a central line and variable thickness function.

The **medial axis transform** [12,13] formulates a shape as the radial offset of its **medial axis** (curve skeleton). The **pearling** approach to image segmentation [14,15,16], recently introduced by Whited, Rossignac, Slabaugh, Fang, and Unal, computes an approximation of the medial axis transform of a grey-level image in realtime by pushing a disk of variable radius along a path made of desired grey-level pixels.

The **ball morph**, recently introduced by Whited and Rossignac [15,18], expresses one shape as the ball offset of another and uses this formulation to measure and optionally exaggerate local discrepancies between similar shapes or to animate the morph between them.

We say that two shapes are **offset-compatible**, when each one can be expressed as the variable distance offset of the other. The compatibility conditions depend on which formulation of variable offset is used. Chazal, Lieutier, and Rossignac [8,9] have provided sufficient and tight compatibility conditions for the orthogonal offset in terms of the minimum feature size of both sets and of their Hausdorff distance [19]. Subsequently, Chazal, Lieutier, Rossignac, and Whited [11] have shown that the ball offset allows for a similar, but less constraining compatibility condition.

Whited and Rossignac [20] have proposed a set theoretic formulation of variable radius **relative blending** where the blending radius is defined locally relative to a second (control) shape as the radius of the ball that is tangent to both shapes.

In this paper, we review these recent results and explore their interactions. In Section 2, we start with a brief review of the terminology and key concepts that will be used throughout the paper. In Section 3, we discuss constant radius offsetting and its applications to blending and tightening. In Section 4, we present the details of the formulations of the three variable distance offsets and compare their results. In Section 5, we discuss the computation of the offset distance field and mapping between two curves. In Section 6, we discuss offset compatibility conditions. In Section 7, we discuss variable radius relative blending. In Section 8, we discuss the Pearling segmentation of ball-swept structures. In Section 9, we discuss shape morphing and averaging.

2 Review of Key Concepts and Terminology

In this section, we review the basic assumptions and define a few standard, and less standard terms and concepts used throughout the paper.

Topological Domain

For simplicity, we restrict our attention to closed, bounded, manifold, and simply connected planar regions that are hence homeomorphic to a closed disk. We use the letters S and R to denote such regions and the letters P and Q to denote their respective (single loop) bounding curves.

P is a Jordan curve and decomposes the plane into three regions: P , the interior iP of P , and the exterior eP of P . The topologically closed set S is the union of P with iP .

Smoothness and Discrete Models

Some of our theoretical results assume that the bounding curve P of the region of interest S is smooth, and thus has a unique normal outward-pointing direction $N_p(p)$ at each point p of P and that this direction varies continuously along the curve (i.e., it satisfies the Lipschitz condition, see [9, 11] for details).

Nevertheless, most of the techniques based on these theoretical results have been successfully applied to shapes bounded by polygonal curves or to discrete models composed of black&white or even grey-level pixels of a regular grid (by using a local estimation of P and of its normal).

Distance and Closest Projection

For simplicity, we will denote by pq the **vector** from point p to point q , which is sometimes written as the “difference” $q-p$ between points.

The **distance** $d(p,q)$ between points p and q may be expressed as the norm $\|pq\|$ of the vector between them.

The **distance** $d(q,P)$ between a point p and a curve P is the minimum of the distances $d(p,q)$ between p and all points q of Q .

The **distance** $d(P,Q)$ between two curves, P and Q , is the minimum of $\|p-q\|$ for all pairs of points p in P and q in Q .

The **closest projection** $c(q,P)$ of point q on curve P is the set of points p such that $d(p,q) = d(q,P)$.

Note that these definitions are not restricted to curves, but also apply to more general sets P and Q .

However, note that when P and Q are smooth curves, if $p = c(q,P)$ and $d(p,q) \neq 0$, then the line segment (p,q) is orthogonal to P at p . Furthermore, if $d(P,Q) \neq 0$, then there exists a point p in P and a point q in Q such that the line segment (p,q) is orthogonal to P at p and to Q at q and that $d(p,q) = d(P,Q)$.

Cut and Medial Axis

Most points q have a single point closest projection $c(q,P)$ on P . The noteworthy exceptions play an important role in shape analysis.

The **cut** $C(P)$ of curve P is the set of points q for which $c(q,P)$ is not a single point. Since the curve P is smooth, it has no common point with $C(P)$.

We differentiate between the interior and the exterior part of the cut. The portion $C_i(P)$ of $C(P)$ in S is called the **interior cut** and is the **medial axis** [12] of S . $C_e(P)$ denotes the **exterior cut** that lies in the **complement** \bar{S} of S .

Reach, Regularity, and Minimum Feature Size

A point q in the complement of $C(P)$ may be represented by its closest projection $p = c(q,P)$ on P and of the signed orthogonal offset distance $d(q) = pq \cdot N_p(p)$. The **range**, $[r_i(p), r_e(p)]$, of p is the set of values $d(p)$ for all points q such that $p = c(q,P)$. For example, the range for each point on a circle of radius r is $[-r, \infty]$. The **interior reach** $-r_i(p)$ is the distance $d(p, C_i(P))$ from p to the medial axis of P . the **exterior reach** $r_e(p)$ is the distance $d(p, C_e(P))$ from p to the exterior cut of P .

The **regularity** [5] $r(p)$ of p with respect to P (sometimes also called the **local feature size** or **reach**) is the minimum, $\min(-r_i(p), r_e(p))$, of its two reaches. It is the distance from p to the cut.

Note that, to simplify notation, we assume that these measures are implicitly defined with respect to the curve P . Hence, instead of $r_p(p)$, we simply write $r(p)$.

The **regularity** $r(P)$ of the whole curve P , also called the **minimum feature size** $\text{mfs}(P)$ of P , is the minimum of the regularity of its points. Hence, we say that curve P is **r-regular** if $r(P) = r$.

3 Constant Radius Offsetting, Blending, and Tightening

In this section, we discuss operations defined in terms of balls of a fixed radius (a specialization of the standard Minkowski operations) and discuss their applications.

Minkowski Sum, Closing, and Opening

The **Minkowski sum** (also called Minkowski addition or **dilation**) $S+B$ of two sets is the set of points $s+ob$, where point o is a chosen origin, s is a point of S , b a point of B and where ob denotes the vector from point o to point b . Equivalently, $S+B$ may also be defined as the union of sets $B+s$, which are translations of B by vector os , for all points s in S . Hence, $S+B$ may be called the **translational sweep** of the **structuring element** B along S .

The **Minkowski difference** (also called Minkowski subtraction or **erosion**) $S-B$ is defined as the intersection of the sets $S-b$, for all points b in B , which are translations of S by vector bo . Equivalently, it is the set of points x such that B translated by ox lies in S .

Minkowski **closing** of set S by the structuring element B is defined as $(S+B)-B$ and its Minkowski **opening** is defined as $(S-B)+B$.

Minkowski operations are the basic components of **mathematical morphology**. They were developed by Georges Matheron and Jean Serra [1,2].

Constant Radius Offsets

By choosing the structuring element B to be a ball of radius r centered at the origin, Rossignac and Requicha [3] have defined the positive **r-offset** (**grow** or **dilation**) $S \uparrow^r$

of set S by a constant distance (also called radius) r as the set of points at distance less or equal to r from S . Equivalently, $S^{\uparrow r}$ is the union of balls with radius r and center in S . Similarly, they define the negative r -offset (**shrinking** or **erosion**) $S^{\downarrow r}$ of S by a constant distance r as the set of points at a distance of more than r from the complement \bar{S} of S . Both are illustrated on Fig. 1.

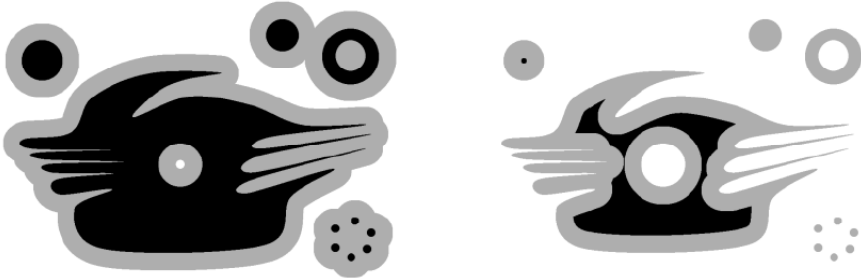


Fig. 1. The shape S is shown black (left). The material $S^{\uparrow r} - S$ added by dilation is shown grey (left). On the right, the shrunk set $S^{\downarrow r}$ is shown in black and the material $S - S^{\downarrow r}$ removed by erosion is shown in grey.

In their definitions, if S is topologically closed (i.e., contains its boundary), then $S^{\uparrow r}$ and $S^{\downarrow r}$ are also closed.

Hausdorff Distance

The Hausdorff distance, $h(P, Q)$ between two curves (or more generally two sets), P and Q , is the minimum value of r such that $P \subset Q^{\uparrow r}$ and $Q \subset P^{\uparrow r}$. It measures the maximum distance from a point of either set to the other. It is often used to measure the discrepancy between two similar sets, and verifies $h(P, Q) \Leftrightarrow P = Q$.

Filleting and Rounding

Rossignac and Requicha [4] have defined two **constant-radius blending** operations for blending (i.e., rounding) all the concave or all the convex corners of a shape S .

The **filleting** $F_r(S)$ of S (also called its **r -closing** or **r -fill**) is defined as $F_r(S) = (S^{\uparrow r})^{\downarrow r}$. It is the set of points of S that cannot be reached by an open ball B of radius r that is disjoint from S . Hence, $F_r(S)$ has no sharp concave corners and no narrow cracks or holes.

The **rounding** $R_r(S)$ of S , (also called its **r -opening** its **r -round**) is defined as $R_r(S) = (S^{\downarrow r})^{\uparrow r}$. It is the set of points of S that can be reached by a closed ball of radius r in S . Hence, $R_r(S)$ has no sharp convex corners and no narrow protrusions or constrictions.

Note that $R_r(S) \subset S \subset F_r(S)$, but in general $R_r(S) \neq F_r(S)$, see Fig. 2. In fact, as r grows, $R_r(S)$ shrinks and $F_r(S)$ grows.



Fig. 2. On the left, we show the result $R_r(S)$ of rounding in black and the material $S - R_r(S)$ removed by the rounding operation in grey. On the right, we show S in black and the material $F_r(S) - S$ added by filleting in grey.

Williams and Rossignac [5] argue that a set S is **r-regular** if $R_r(S) = F_r(S)$. In other words, S is r-regular if it is equal to its r-closing and to its r-opening.

Some sets cannot be made r-regular by applying any arbitrary combination of r-closing or r-opening, as each application of r-closing leaves sharp convex corners, while the application of r-opening leaves sharp concave corners.

The r-filleting and r-rounding operations defined above are biased. $F_r(S)$ removes concave sharp features and grows the object. $R_r(S)$ removes convex sharp features and shrinks the object.

Mortar and Finite-Scale Topological Operators

Williams and Rossignac [5] define the **r-mortar** $M_r(S)$ of S as the filleting $F_r(P)$ (i.e., the r-closing $(P^\uparrow)^\downarrow_r$) of its boundary P (Fig. 3). In computational geometry, $M_r(S)$ would be called the **alpha hull** of P . Note that $M_r(S) = F_r(S) - R_r(S)$. Note that the r-fill of S is the union $F_r(S) = R_r(S) \cup M_r(S)$ of its rounding with its mortar.

Hence, as shown in Fig. 4, computing the r-mortar “inflates” the boundary of S and decomposes space into three disjoint sets: the **r-interior** $R_r(S)$, the **r-boundary** $M_r(S)$, and the **r-closure** $F_r(S)$. In the limit, as r tends to zero, these operators converge to the standard topological interior, boundary, and closure operators. But for a finite r , they provide an interesting variable-scale version of these topological operators.

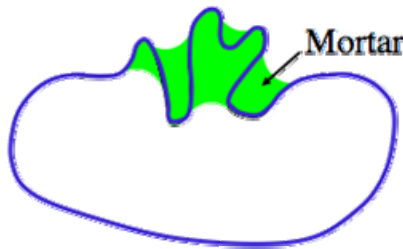


Fig. 3. The Mortar (shaded region) of curve P is the space not reachable by an open ball disjoint from P

The r -mortar is one-dimensional (i.e. equal to P) at **r -regular points** p of P , where the regularity $r(p) \geq r$. The r -mortar is a two-dimensional region around the irregular portions of P where $r(p) < r$.



Fig. 4. From left: The set S . The r -rounding $R_r(S)$ in black and the difference $S - R_r(S)$ in grey. S in black and the difference $F_r(S) - S$ in grey. $R_r(S)$ in black and $M_r(S) = F_r(S) - R_r(S)$ in grey.

Stability

Williams and Rossignac [5] define the **stability** of a point q with respect to set S as the smallest value of r for which q belongs to $M_r(S)$. The measure of stability throughout space, not just on the boundary P of S , (see Fig. 5) is a powerful mathematical morphology tool for analyzing how S is imbedded in space. The information it provides cannot be extracted from a topological characterization of S not from a differential analysis of its boundary.

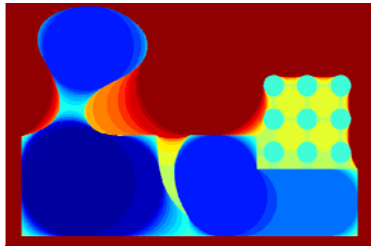


Fig. 5. The stability of a point is proportional to the darkness of the color (capped outside)

Blending Combinations and the Mason Filter

The **blending combinations** $F_r(R_r(S))$ and $R_r(F_r(S))$ tend to eliminate both concave and convex sharp features and often produce r -regular shapes, but in general retain a bias: $F_r(R_r(S))$ is often a strict subset of S and S is often a strict subset of $R_r(F_r(S))$.

One may prefer an unbiased blending operator that is **symmetric** (so that applying it to S or to the complement of S will yield the same result).

Williams and Rossignac [5] show that $R_r(S) \subseteq R_r(F_r(S)) \subseteq F_r(S)$ and $R_r(S) \subseteq F_r(R_r(S)) \subseteq F_r(S)$ and that the application of arbitrary combinations of r -closing and r -opening only alter S in the full dimensional portion of its r -mortar.

Hence, they propose the **Mason filter** [5], which decomposes the two-dimensional portion of the r -mortar into connected components and in each component C of the r -mortar of S , replaces S by either $F_r(R_r(S))$ or $R_r(F_r(S))$: they pick the combination that alters the smallest (in area) portion of C .

Note that Mason acts symmetrically on S and on its complement. Williams and Rossignac show that the output of Mason is guaranteed to have a smaller symmetric difference with the original shape than either $F_r(R_r(S))$ or $R_r(F_r(S))$. Its application to a binary image is shown in Fig. 6.

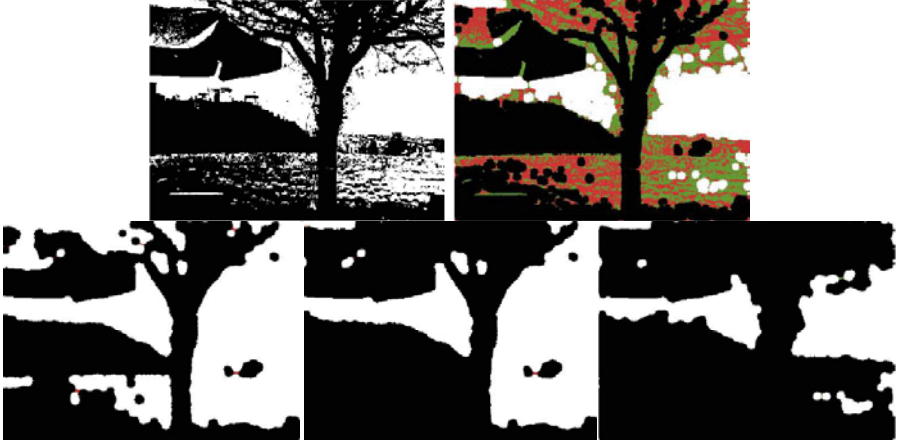


Fig. 6. The original image (top left) is decomposed (top right) into its r -rounding (black), the complement of its r -filleting (white), and its r -mortar. We compare the result of $F_r(R_r(S))$, shown bottom left, the result of the r -Mason filter (bottom center), and the result of $R_r(F_r(S))$.

As a shape simplification operator, Mason offers advantages over other smoothing or vertex decimation filters.

- 1) It may be used to “regularize” (i.e., remove sharp corners, holes, speckles, thin branches or constrictions) the shape at different scales (Fig. 7).



Fig. 7. The original shape (left) and the results (left-to-right) of applying Mason with increasing values of r

- 2) It is expressed in terms of set-theoretic morphological operators and is hence independent of the representation used for S or P .
- 3) It is self-dual, meaning that $\text{Mason}(S) = !(\text{Mason}(!S))$, where $!X$ denotes the complement of set X , and treats positive and negative space symmetrically.
- 4) It may be viewed as a new tolerance zone that confines the effects of shape simplification to irregular regions, where the boundary has high curvature or where two distinct portions of the boundary are close to each other.
- 5) It preserves the portions of the boundary of S that are regular at the desired scale r .

Tightening

Improving on Mason, Williams and Rossignac propose the **r-tightening** [6,7], which alters S by tightening its boundary in its r -mortar. The r -tightening minimizes the arc-length of P while keeping P inside the r -mortar (Fig. 8). Topological changes may be necessary to produce a new boundary that is smooth (i.e., for which the radius of curvature is never less than r). The computation of the r -tightening is analogous to the computation of the shortest path in a corridor.



Fig. 8. The original shape S (left). Its mortar (grey) and the tightened boundary (center). The tightening of S (right).

Example of tightening in 2D and 3D are shown in Fig. 9 and a comparison with Mason in Fig. 10.

4 Variable Distance Offset Formulations

In this section, we explore offsetting formulations where the offset distance d varies along P . We say that d is the **offset distance field**.

Without loss of generality, we assume that P is oriented (for example, clockwise). With each point p of P , we associate an **offset distance** $d(p)$. The curve obtained by offsetting P by the distance field d will be denoted P^d .

We can also compute the unit **tangent** vector, $t(p)$, and unit outward **normal** vector, $n(p)$, to P at p . For simplicity, we will omit the reference to p in these expressions and say that each point p of P is associated with a distance value d , and a **local ortho-normal frame** $\{p, t, n\}$. We will use the notation d' for the **derivative** of the distance function d at p with respect to an arc-length parameterization of P .

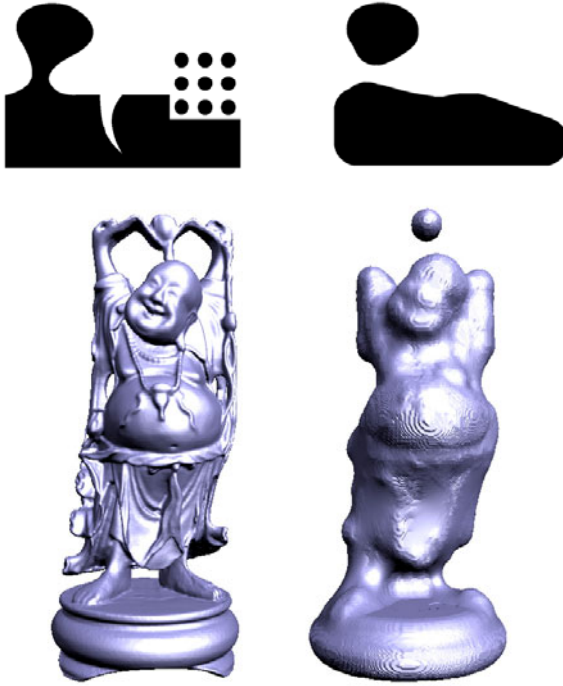


Fig. 9. Original sets (left) and their tightenings (right)



Fig. 10. Top: Original shape S (left) and its mortar (right). Middle: $Rr(Fr(S))$ and $Fr(Rr(S))$. Bottom: r -Mason (left) and r -tightening (right) of S .

For simplicity, we define the **offset** p^d of p in this local frame as $p^d = p + xt + yn$. This is a convenient formulation for computing the offsets of samples p along P .

We propose three different offsets. We provide here the local constructions of the offset point p^d . Compatibility conditions are discussed further in the paper.

Orthogonal Offset

To differentiate it from other offset formulations, the **orthogonal offset** P^d of curve P by a distance field d is denoted by $O_d(P)$. The orthogonal offset point p^d and the normal vector n^d to P^d at p^d may be computed as:

$$p^d = p + d n \quad \text{and} \quad n^d = (-d' t + n) / \sqrt{(d')^2 + 1}. \tag{1}$$

The value of d does not need to be positive, hence the orthogonal offset curve may be on the left or on the right of P and may cross P when $d=0$.

The line segment (p, p^d) is orthogonal to P at p , but not necessarily to P^d at p^d .

Radial Offset

The **radial offset** P^d of curve P by a distance field d is denoted by $R_d(P)$. The radial offset point and its normal may be computed as:

$$n^d = (-d' t + \sqrt{(1 - d')^2} n) \quad \text{and} \quad p^d = p + d n^d. \tag{2}$$

In portions where d is negative, the radial offset curve will lie on the right of P .

Note that the segment (p, p^d) is orthogonal to P^d at p^d , but not necessarily to P at p .

The radial offset is a portion of the boundary of the region swept by a disk with center p and radius $d(p)$ as p moves along P . As such it is the subset of the envelop [10] of the family of these disks (Fig. 11).

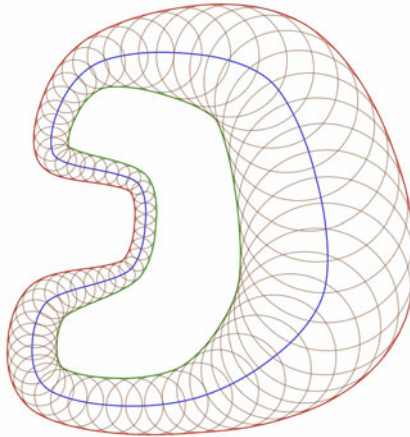


Fig. 11. We show three curves: I (inner), P (central), and O (outer). O is the radial offset by distance field r of P. It is the envelop of a family of circles with centers on P. I is the radial offset of P by the negative $-r$ of the distance field. Note that P is the orthogonal offset by r of I and that O is the ball offset by $2r$ of I.

Ball Offset

The **ball offset** (or tangent ball offset) P^d of curve P by a distance field d is denoted by $B_d(P)$. The ball offset is defined as the composition of the other two offsets: $B_d(P) = R_r(O_r(P))$ with $r=d/2$.

Note that a point p^f of $O_r(P)$ inherits the distance $d/2$ from the corresponding point p on P . However, the derivative r' of that distance with respect to the arc-length parameterization of $O_r(P)$ is in general not equal to $d'/2$. It may be expressed exactly using the curvature of P at p or approximated from a sampling of $O_r(P)$.

The ball offset is a portion of the boundary of the region swept by a disk with radius $d(p)/2$ that is tangent to P at p , as p moves along P . As such, it is the subset of the envelop of the family of these disks.

Comparison and Properties of the Three Offsets

In the special case where $d(p)$ is a constant, $B_d(P) = R_d(P) = O_d(P)$ and hence all three are equal to the constant distance offset discussed earlier. The local disparity between these three offsets increases with the derivative of d .

We compare the three offset formulations in Fig. 12, illustrating their construction at one particular point p of P .

Then P is smooth, the three constructions proposed above establish a homeomorphism between P and P_d . We may use this correspondence to transfer the distance field, as we did above for the construction of the ball offset, even though the derivative of the distance field may not be transferable directly, as offsetting is in general not length preserving. Hence, writing $R_{f(d)}(O_d(P))$, we imply that if a point p of P is associated with an offset distance d , then its image p^d of p by O^d is associated with an offset distance of $f(d)$, where f is some function of d .

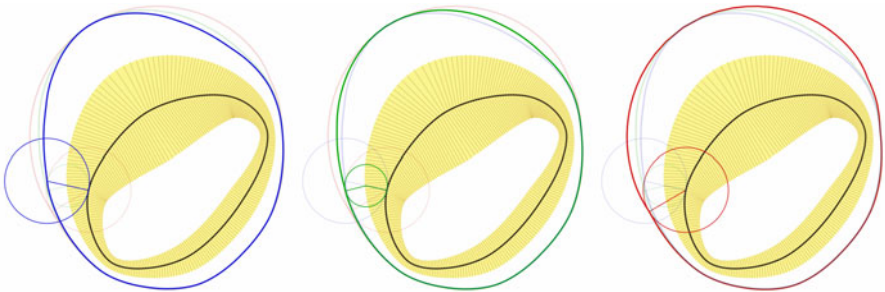


Fig. 12. In all three images, we show the same black curve P and indicate its distance field d by the thickness of the shaded region around it (computed as its radial offset). Each curve shows a different offset superimposed on faded versions of the other two for more precise comparison. We show the construction of the offset p^d for a particular point p of P . The normal offset is shown left. The ball offset is shown center. The radial offset is shown right. Note that the radial offset is the furthest from P and that the normal offset is the closest.

We observe that R_d is the inverse of O_d (Fig. 13 left) in the following sense:

$$R_{-d}(O_d(P)) = P \quad \text{and} \quad O_{-d}(R_d(P)) = P \quad (3)$$

It follows that if $O_d(P) = Q$ then $R_{-d}(Q) = P$.

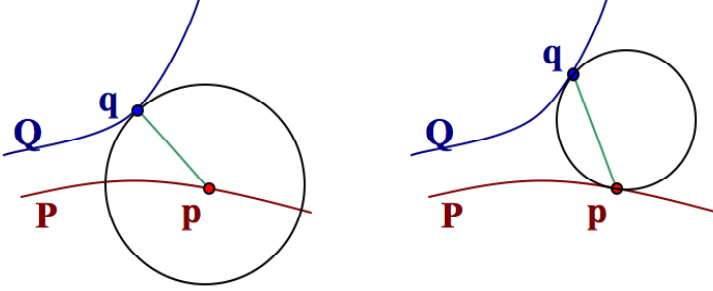


Fig. 13. Left: point q is the orthogonal offset by d of p with respect to P and p is the radial offset by $-d$ of q with respect to P . Hence, $Q=O_d(P)$ and $P=R_{-d}(P)$. Right: q is the ball offset of p by d with respect to P and p is the ball offset by $-d$ of q with respect to Q . Hence, the ball offset is its own inverse.

Similarly, B_d is its own inverse (Fig. 13 right):

$$B_{-d}(B_d(P)) = P \quad (4)$$

It follows that if $B_d(P) = Q$ then $B_{-d}(Q) = P$.

5 Computation of the Offset Distance Field between Two Curves

We are given two smooth curves, P and Q , and want to express one as the offset of the other. The computation of the distance field depends on which offset is desired.

Orthogonal Map and Offset

For an orthogonal offset [8,9], the distance field d that satisfies $Q = O_d(P)$ may be computed at each point p by constructing a line L that is orthogonal to P at p , and by computing the intersections of L with Q . We distinguish two kinds of intersections: **kissings**, where L osculates Q (i.e., is in tangential contact to Q without crossing it) and **crossings**, where L crosses Q . Note that the number of crossings is always even. Furthermore, the orientation of the crossings alternates along L : as L enters the region R bounded by Q it crosses Q from the left (with respect to the orientation of Q), as it leaves R , it crosses Q from the right.

If for some point p there are no such crossings, then Q cannot be expressed as the orthogonal offset of P .

If there are crossings at each point p , we find the nearest crossing from p in each direction along L . We select the crossing q that has the correct orientation: If L crosses P at p from the left, then L must also cross Q at q from the left. The distance d

associated with p is the dot product $(p-q) \cdot n$, i.e., the signed distance $\|p-q\|$. However, if this segment (p,q) of L contains a kissing or if L is tangential to Q at q , we declare that Q cannot be expressed as the orthogonal offset of P . In these cases, we conclude that P and Q are not O -compatible.

Radial (Closest-Projection) Map and Offset

For a radial offset, we associate with each point p of P the distance $d(p,Q)$ from p to the closest point q on Q . Hence, we map p to its closest projection q on Q .

Ball Map and Offset

For a ball offset [11], we consider the moat X , which is the symmetric difference (XOR), $S \otimes R$ between the region, S and R , bounded respectively by P and Q . Then, we compute the medial axis transform of X .

If X has bifurcations or kinks (points where the normal is discontinuous), we declare that P and Q are not **B-compatible**.

Otherwise, the medial axis defines a family of disks in X that are each tangent to P at exactly one point p and tangent to Q at exactly one point q (Fig. 14). We say that q is the image of p by a **ball map** [11], from P to Q and vice versa. We associate with p and with q the diameter of that tangent disk, but with opposite signs. The sign associated with p is positive when p lies inside R .

The distance field d assigned to P satisfies $B_d(P) = Q$. We also have $B_{-d}(Q) = P$.

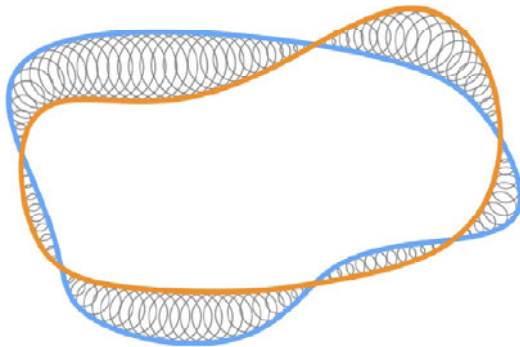


Fig. 14. When the maximal disks in the moat between the two curves P and Q touch each curve at a single point, they define the correspondence (ball-map) between a point p of P and a point q of Q . Furthermore, the diameter of the touching ball is the ball offset distance associated with these contact points.

6 Compatibility

We say that a distance field d is **X-compatible** with a curve P , where ‘ X ’ stands for ‘ O ’, ‘ B ’, or ‘ R ’, when the X -offset of P by d is free from cusps and self-intersections.

Incompatibilities may be produced **locally**, when for example the orthogonal offset distance exceeds the local radius of curvature (this is sometimes called the fish-tail and illustrated in an inward offset of an ellipse) or when the derivative of the distance field of a radial offset exceeds 1 (the disk of radius d at p is contained inside the disk of radius $d+\epsilon d'$ centered at a nearby point $p+\epsilon t$, for an infinitely small ϵ).

Incompatibilities may also be produced **globally**, where the images on P^d of two disjoint parts of P intersect.

We say that two curves P and Q are **X-compatible** if each one can be expressed as the X -offset of the other (i.e. if a suitable X -compatible distance field may be found).

Note that, because an orthogonal offset is the inverse of a radial offset, O -compatibility implies B -compatibility and vice versa.

For example, Fig. 15 shows two curves that are B -compatible, but are not O -compatible, and hence not R -compatible either.

Chazal, Lieutier, and Rossignac [8,9] prove that two smooth curves P and Q are O -compatible (and hence also R -compatible) if

$$h(P,Q) < (2 - \sqrt{2}) \min(r(P),r(Q)) \tag{5}$$

This sufficient, although not necessary condition bounds the disparity (measured in terms of Hausdorff distance) as a function of the minimum feature sizes (or regularity) of the two curves. They also show that the constant $(2 - \sqrt{2})$ is tight by producing an example of two curves that are not O -compatible (because they intersect at right angles) and for which we have $h(P,Q) = (2 - \sqrt{2}) \min(r(P),r(Q))$.

Chazal et al. [11] prove that two smooth curves P and Q are B -compatible if

$$h(P,Q) < \min(r(P),r(Q)) \tag{6}$$

This bound is also tight, but is less constraining, since it allows a greater disparity between the curves for the same feature size. Hence, we conclude that B -compatibility is easier to achieve than O -compatibility.

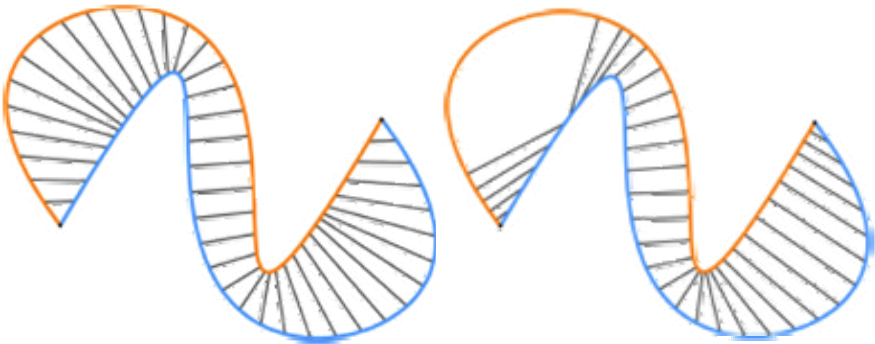


Fig. 15. The two curves are B -compatible (i.e., each one is the ball offset of the other). The correspondence (ball map) is shown on the left as line segments joining points p to corresponding points q . But the two curves are not O -compatible, nor R -compatible, as shown on the right.

7 Variable Radius Relative Blending

When blending (i.e., removing sharp features of) a shape S , one may want the radius of the blending ball to vary along the boundary P of the shape S . This objective raises two challenges:

- 1) How to define the radius function
- 2) How to define the result of the blend

Whited and Rossignac [20] propose their variable-radius **relative blending** approach where the radius field used to blend S is defined as half the ball offset distance from the boundary P of S to a control curve Q .

Their approach removes the portions of P that are incompatible with Q and replaces them with smoothly connecting, circular-arc blends, as shown in Fig. 16. This operation corresponds to the removal of dangling branches of the medial axis. The result is not truly compatible with the control curve, since the maximal ball centered at the bifurcation of the original medial axis has more than one contact point with it, but is pseudo-compatible and hence, one can establish a ball map between the two curves and can express the new rounded curve as the ball offset of the control curve.

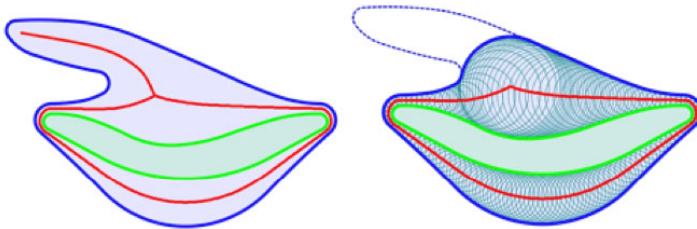


Fig. 16. The outer-curve (left) has a protrusion (top) that is incompatible with the inner (mouth-like) control curve. This incompatibility is indicated by a bifurcation of the medial-axis of the symmetric difference between the two curves. On the right, the protrusion has been removed (dotted lines) and was replaced by a circular arc.

Furthermore, Whited and Rossignac [20] provide a set-theoretic formulation of the desired result in terms of S , the union C of the maximal balls in the symmetric difference between P and R that touch both P and Q , and the set M bounded by the trimmed medial axis loop (completed by adding the intersections and overlap between P and Q). The relative blending $B_R(S)$ of set S with respect to set R is

$$B_R(S) = (S \cap (M \cup C)) \cup (M - C) \tag{7}$$

They show that this approach may be used in three dimensions to simultaneously blend the concave and convex features of a 3D shape with variable radius fillets and rounds (Fig. 17).

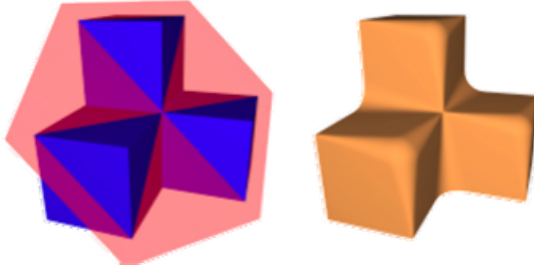


Fig. 17. The semitransparent plane (left) is used as a control surface to blend a complex corner of a polygonal solid. The resulting shape (right) contains variable-radius blends.

8 Pearling Segmentation

Whited et al. [14,15] have proposed an interactive technique (called **Pearling**) for segmenting strokes in images and also tubular structures in 3D medical data sets. For example, to trace a road or artery in an image (Fig. 18), they start from a user-provided position and direction on the road and create, one at a time, a string of balls along the road. The position and radius of each ball are adjusted using a few iterations so that the ball is not far from the previous one and so that its center is mostly filled with “good” pixels that have “road-like” colors and that a small periphery on opposite



Fig. 18. A particular street is traced by Pearling in real-time. Upon request, the tracing may expand to branching streets.

sides is filled with “bad” pixels that repel the center of the ball. To define “good” and “bad” pixels, the user selects sample good and bad regions by painting over a portion of the road and one or more portions of its surrounding.

The construction is interactive (the road is traced as fast as a rough centerline can be drawn by the user) and can track bifurcation. A smooth curve that interpolates the centers of the balls may be viewed as an approximation of the **centerline** of the road. Treating it as the medial axis leads to the formulation of the segmented region in terms of its medial axis transform. Hence, away from the bifurcation points, the left and right borders of the road are the radial offsets of the centerline and one border is the ball-offset of the other. This observation is important for subsequent processing of the results, for example to remove the bulges associated with possible, but not traced bifurcations.

In 3D, Pearling has also been used [16] to trace human vasculature (Fig. 19) from discrete medical scans. A similar approach was developed for tracing tubes in solids represented by triangle meshes [21,22].

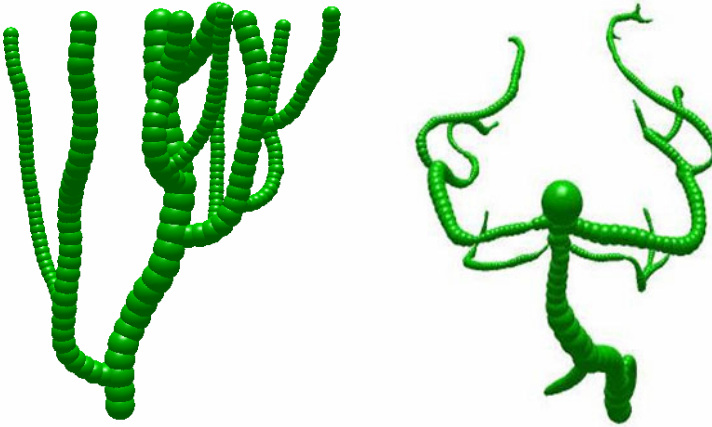


Fig. 19. Tubular structures in human anatomies segmented by Pearling from medical scans

9 Shape Morphing and Midarc Averaging

In their **ball morph** approach to in-betweening, Whited and Rossignac [17,18] propose to use the ball map correspondence [11] to define a curved trajectory from each point of curve P to a single point of a B -compatible curve Q . The trajectory is a circular arc that is orthogonal to p at P and to q at Q (Fig. 20 left).

They establish these ball map correspondences and trajectories for a coordinated sampling of P and Q , sample the trajectories uniformly, and finally join each set of corresponding samples by a closed-loop curve (Fig. 20 right). These curves may be used as **frames** of an in-betweening animation (Fig. 21). More importantly, this formulation also defines a **continuous animation model**.

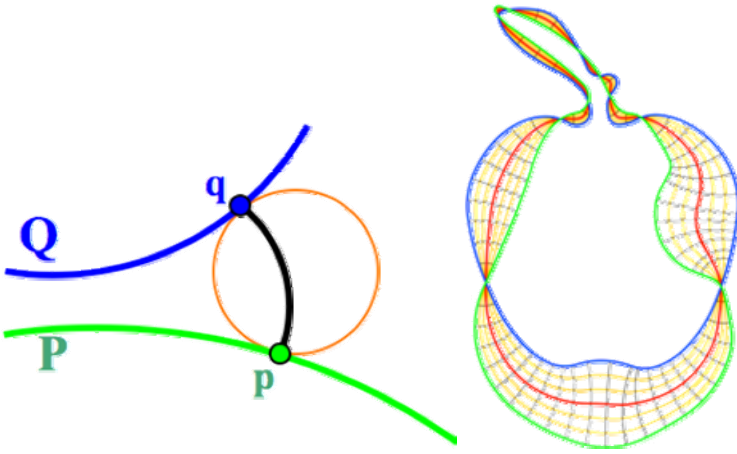


Fig. 20. The circular trajectory from p to q is fully defined by the corresponding maximal disk

The halfway frame (which joins the mid-course point on each arc) provides a new definition of the **average** of two shapes. We call it the **midarc average**. It differs from the medial axis of the gap between the two curves (Fig. 22 left).

This formulation may be extended to produce a new type of centerline (with bifurcations) of a single shape. We call it the **midarc axis**. We contrast it (Fig. 22 right) with prior definitions: Blum's Medial Axis [MAT Blum], Layton's PISA [23], and Asada and Brady's Smooth Locus of Symmetry [24].

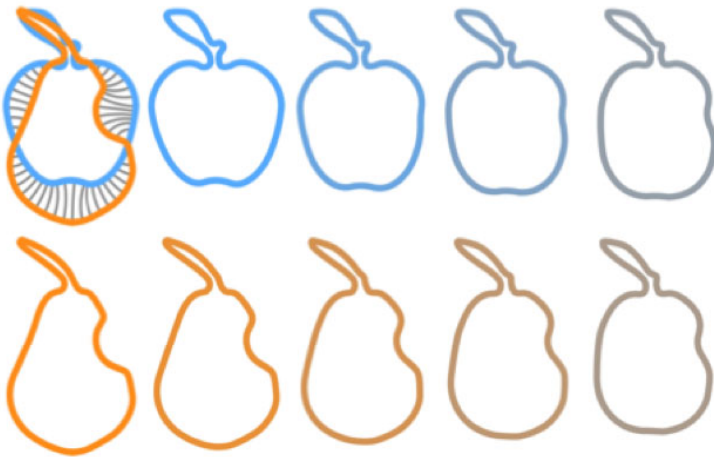


Fig. 21. Trajectories (top left) and frames of a Ball morph animation between an apple and a pear

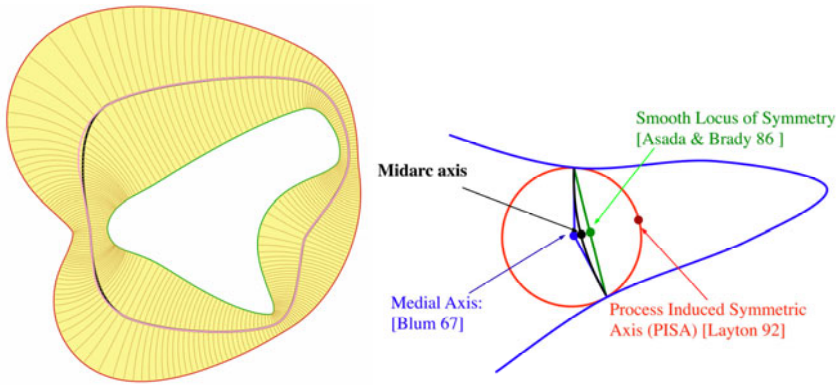


Fig. 22. The midarc average of two curves is the mid-arc axis of the gap between them. It is superimposed (left) over the darker medial axis. The construction of the midarc axis is contrasted (right) with other definitions of centerlines.

10 Conclusion

We have briefly reviewed a variety of ball-based techniques for analyzing, transforming, comparing, and morphing shapes and have contrasted them with closest-projection techniques. Specifically, ball-based techniques may be used to blend shapes so that they are B-compatible and to create a correspondence between B-compatible shapes which can be used to model one shape as the ball offset of another, to compute weighted averages of shapes, and to produce in-betweening animations that smoothly morph from one shape to another.

Several challenges remain.

One challenge is the closure of the domain. In two dimensions, these techniques have been implemented for discrete models with regularly spaced samples (pixels) and with ordered samples spaced along the curve, for polygonal models, and for piecewise-circular (PCC) [25] representations of the curves. The first two do work in practice when the sampling density is sufficient, but are theoretically incorrect. The latter is theoretically correct and efficient, but may produce derived offset curves (that are not PCC and hence must be approximated by a PCC if one wishes to offset them again).

Another challenge is the extension of these techniques to three dimensions. Using efficient (hardware-assisted) algorithms for computing offsets and distance fields permits to implement most of the techniques discussed here on discrete (voxel) and point cloud representations [26] and also on triangle-meshes [11]. However, the formulation of the tightening of a three-dimensional shape is not as straightforward, because the minimization of the surface area within the 3D mortar does not in general yield an acceptable solution. This theoretical extension is the topic of the forthcoming PhD dissertation of student Jason Williams.

A third challenge is the evaluation of the practical and perceptual benefits of using ball offsets rather than orthogonal or radial offsets for drawing variable thickness curves and of using the midarc axis rather than the medial axis as a skeletal abstraction of a shape.

A fourth challenge is the extension of these approaches to more than two curves or more than two surfaces. This aspect is currently explored by the author in collaboration with Drs. Raimund Seidel and Brian Wyvill.

Finally, one may wish to investigate further the relation between the variable radius relative blending and the variable distance offsets discussed here and the adaptive neighborhood operations used in mathematical morphology [27]. Similarly, one may wish to investigate the further relation between the Mason and Tightening filters discussed above and the Alternating Sequential Filters used in mathematical morphology [28].

Acknowledgements

The results presented here were developed in collaboration with the author's Ph.D. students Jason Williams and Brian Whited (now at Disney Animation), and with colleagues Frédéric Chazal (INRIA), André Lieutier, (Dassault Systmes), Greg Slabaugh (formerly at Siemens), and Tong Fang and Gozde Unal (both from Siemens). Nick Patrikalakis (MIT), Chris Hoffman (Purdue), and Bruno Levy and Nicolas Ray (INRIA) have also provided valuable input to some aspects of this work.

References

1. Serra, J.: *Image Analysis and Mathematical Morphology*. Academic Press, New York (1982)
2. Matheron, G.: *Random Sets and Integral Geometry*. J Wiley Sons, New York
3. Rossignac, J., Requicha, A.: Offsetting Operations in Solid Modelling. *Computer-Aided Geometric Design* 3, 129–148 (1986)
4. Rossignac, J., Requicha, A.: Constant-Radius Blending in Solid Modeling. In: *ASME Computers In Mechanical Engineering (CIME)*, vol. 3, pp. 65–73 (1984)
5. Williams, J., Rossignac, J.: Mason: Morphological Simplification. *Graphical Models* 67(4), 285–303 (2005)
6. Williams, J., Rossignac, J.: Tightening: Curvature Limiting Morphological Simplification. In: *ACM Symposium on Solid and Physical Modeling (Sketch)*, pp. 107–112. MIT, Cambridge (June 2005)
7. Williams, J., Rossignac, J.: Tightening: Morphological Simplification. *International Journal of Computational Geometry & Applications (IJCGA)* 17(5), 487–503 (2007)
8. Chazal, F., Lieutier, A., Rossignac, J.: Orthomap: Homeomorphism-guaranteeing normal projection map between surfaces. In: *ACM Symposium on Solid and Physical Modeling*, pp. 9–14. MIT, Cambridge (June 2005)
9. Chazal, F., Lieutier, A., Rossignac, J.: Normal map between normal-compatible manifolds. *International Journal of Computational Geometry & Applications (IJCGA)* 17(5), 403–421 (2007)
10. Monge, G.: *Applications de l'analyse a la geometrie*, 5th edn., Bachelier, Paris (1894)
11. Chazal, F., Lieutier, A., Rossignac, J., Whited, B.: Ball Map: Homeomorphism between compatible surfaces. *International Journal of Computational Geometry and Applications (CG&A)* 20(3), 285–306 (2010)
12. 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)

13. Foskey, M., Lin, M., Manocha, D.: Efficient computation of a simplified medial axis. In: ACM Symposium on Solid Modeling and Applications, pp. 96–107 (2003)
14. Whited, B., Rossignac, J., Slabaugh, G., Fang, T., Unal, G.: Pearling: Stroke segmentation with crusted pearl strings. In: The First International Workshop on Image Mining Theory and Applications, IMTA (2008)
15. Whited, B., Rossignac, J., Slabaugh, G., Fang, T., Unal, G.: Pearling: Stroke segmentation with crusted pearl strings. *Journal of Pattern Recognition and Image Analysis (PRIA)* 19(2), 277–283 (2009)
16. Whited, B., Rossignac, J., Slabaugh, G., Fang, T., Unal, G.: Pearling: 3D interactive extraction of tubular structures from volumetric images. In: MICCAI Workshop: Interaction in Medical Image Analysis and Visualization (2007)
17. Whited, B., Rossignac, J.: B-morphs between b-compatible curves. In: ACM Symposium on Solid and Physical Modeling, SPM (2009)
18. Whited, B., Rossignac, J.: Ball morph: Definition, Implementation, and Comparative Evaluation. *IEEE Trans. on Visualization & Computer Graphics* (2011), <http://www.computer.org/portal/web/csdl/doi/10.1109/TVCG.2010.115>
19. Atallah, M.: A linear time algorithm for the Hausdorff distance between convex polygons. *Information Processing Letters* 16, 207–209 (1983)
20. Whited, B., Rossignac, J.: Relative blending. *Journal of Computer-Aided Design (JCAD)* 41(6), 456–462 (2009)
21. Mortara, M., Patane, G., Spagnuolo, M., Falcidieno, B., Rossignac, J.: Plumber: A method for a multi-scale decomposition of 3D shapes into tubular primitives and bodies. In: ACM Symposium on Solid Modeling (2004)
22. Mortara, M., Patane, G., Spagnuolo, M., Falcidieno, B., Rossignac, J.: Blowing Bubbles for the Multiscale Analysis and Decomposition of Triangle-Meshes. *Algorithmica* 38(1) (2004)
23. Leyton, M.: *Symmetry, Causality, Mind*. MIT Press, Cambridge (1992)
24. Asada, H., Brady, M.: The curvature primal sketch. *IEEE Trans. Pattern Anal. Mach. Intell.* 8(1), 2–14 (1986)
25. Rossignac, J., Requicha, A.: Piecewise-Circular Curves for Geometric Modeling. *IBM Journal of Research and Development* 13, 296–313 (1987)
26. Chen, Y., Wang, H., Rosen, D., Rossignac, J.: Filletting and rounding using a point-based method. *ASME Design Engineering Technical Conferences, DETC05/DAC-85408* (September 2005)
27. Debayle, J., Pinoli, J.C.: Adaptive neighborhood mathematical morphology and its applications to image filtering and segmentation. In: *European Congress for Image Analysis and Stereology* (2005)
28. Morales, A., Acharya, R.: Alternating sequential filters and multiresolution morphology. In: *IEEE International Conference on Systems Engineering*, pp. 534–538 (1990), doi:10.1109/ICSYSE.1990.203212

Hierarchies and Optima

Jean Serra

Université Paris-Est, 2, Bd Blaise Pascal, 93162l Noisy-le-Grand, France
j.serra@esiee.fr

Abstract. This paper is devoted to hierarchies of partitions, on which all criteria are proved to be connective. Optimisations are addressed by minimizing energies that satisfy the condition of hierarchical increasingness. The optimal cuts through the hierarchies are found. It is shown that many of the classical techniques are variants of what is proposed.

1 Introduction: Optimum Optimorum

Various segmentation algorithms base their approach on a stack of increasing partitions depending on a positive index. The stack, which has been obtained by a first wave of processing, serves as a framework for a second step, sometimes followed by a third one, and aims to lead to a final optimal partition of segmentation. There are several reasons to do so. Some regions of the image under study may require a finer treatment than others. Then one starts from over segmentations which are recomposed [2], [18], [19]. Sometimes, one wants to reduce the size of an image while keeping its major features [16]. Another reason appears with multivariate data, when one has to merge partitions coming from different bands [1]. Sometimes also, the hierarchy is a direct consequence of the partitioning algorithm; a topological watershed, for example, produces a series of edges of a constant value, their saliencies [9] [11].

Most of these algorithms yield a unique final partition, though the meaning of the underlying optimization - if it exists- is rather unclear. A significant advance to clarify this point comes from L. Guigues et Al. [5] who introduced a linear decomposition of some energy over partitions. We will extend below the principle of their approach. After having described the structure of the hierarchies of partitions (section 2), we develop some theory about minimum cuts (section 3), which is illustrated by a few examples in section 4.

2 Hierarchies of Partitions

From now on we suppose that a first step of segmentation already led to a finite hierarchy, i.e. to some finite sequence of increasing finite partitions. We will now intend to reorganize the classes of these partitions for extracting their quintessence, namely a more synthetic partition. Does the chain structure of the initial partitions provide us with particular assets for the second step of optimization?

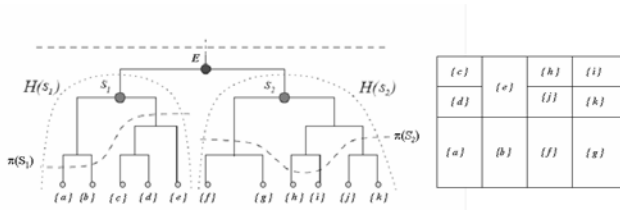


Fig. 1. Left, hierarchical tree; right, the corresponding space structure. S_1 and S_2 are the nodes sons of E , and $H(S_1)$ and $H(S_2)$ are the associated sub-hierarchies. π_1 and π_2 are cuts of $H(S_1)$ and $H(S_2)$ respectively, and $\pi_1 \sqcup \pi_2$ is a cut of E .

2.1 Reminder

Finite hierarchies appeared initially in taxonomy for classifying objects. One can quote in particular the works of J.P. Benzécri [3] and of E. Diday [4]. We owe to the first author the theorem linking ultrametrics with hierarchies, which is involved below in Theorem 5. In image processing, hierarchical structures hold sometimes on stacks of images, but more often of operators, such as the classical families of Gaussian convolutions of D. Marr [7], or again the granulometries of G. Matheron [8]. The hierarchies indicated in the introduction occupy an intermediary position, since they start from a given chain H of segmentations of a fixed function f on set E , i.e. from a stack of scalar or vector images, and this chain is then used as the framework for further operators. Here, for the sake of simplicity, we consider function f and hierarchy H as two starting points, possibly independent. This results in the following definition:

Definition 1. Let \mathcal{D} be the set of all partitions of a finite set E , equipped with the refinement ordering \ll . A hierarchy H of partitions π_i is a finite chain in \mathcal{D} , i.e.

$$H = \{\pi_i, 0 \leq i \leq n, \pi_i \in \mathcal{D} \mid i \leq j \Rightarrow \pi_i \ll \pi_j\}, \tag{1}$$

of extremities the extrema of \mathcal{D} , namely $\pi_0 = \{\{x\}, x \in E\}$ and $\pi_n = \{E\}$.

Denote by \mathcal{S} the set of all classes $S_i(x)$ of all partitions π_i of H , i.e. $\mathcal{S} = \{S_i(x), x \in E, 0 \leq i \leq n\}$. The expression (II) means that at each point $x \in E$ the family of those classes $S_i(x)$ of \mathcal{S} that contain x forms a finite chain \mathcal{S}_x in $\mathcal{P}(E)$, of nested elements from $\{x\}$ to E :

$$\mathcal{S}_x = \{S_i(x), 0 \leq i \leq n\}.$$

The finiteness E is not really necessary. It mainly serves to ensure that the border lengths between classes are finite, but in 2D the Euclidean Voronoi model grants it as well. On the other hand, the search for minimal cuts by an induction on the hierarchical levels (Proposition 7), requires that their number is finite. Remark also that the parameter i of the level can be replaced by any strictly increasing function $t(i)$, which defines another indexing of the hierarchy.

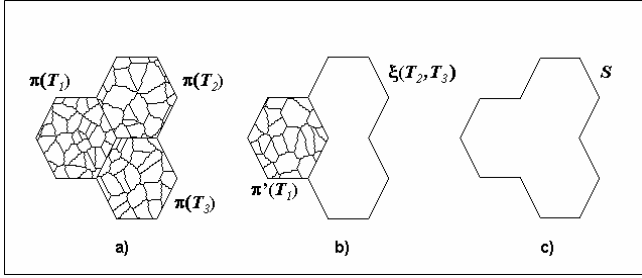


Fig. 2. The three hexagons a) are the three sons T_1, T_2, T_3 of summit S (in c)). The concatenation $\pi(T_1) \sqcup \pi(T_2) \sqcup \pi(T_3)$ of three of their cuts is a cut of S (a). Another cut of S is obtained by taking $T_1 \cup T_2 \cup T_3 = S$ (in c). However, $\pi'(T_1) \sqcup \xi(T_2, T_3)$, in b), is not a valid cut of S .

According to a classical result, a family $\{S_i(x), x \in E, 0 \leq i \leq n\}$ of indexed sets generates the classes of a hierarchy iff

$$\begin{aligned}
 i \leq j \quad \text{and} \quad x, y \in E &\Rightarrow S_i(x) \subseteq S_j(y) & (2) \\
 \text{or } S_i(x) \supseteq S_j(y) \quad \text{or } S_i(x) \cap S_j(y) &= \emptyset.
 \end{aligned}$$

Conventionally, a hierarchy is represented by a tree where each node of bifurcation is a class S . The classes of π_{i-1} at level $i - 1$ which are included in $S_i(x)$ are said to be the sons of $S_i(x)$. Clearly, the sets of the descendants of each S forms in turn a hierarchy $H(S)$ of summit S , which is included in the complete hierarchy $H = H(E)$. Figure 1 depicts two examples of such sub hierarchies, by the two zones $H(S_1)$ and $H(S_2)$ in small dotted lines.

2.2 Cuts in a Hierarchy

Any partition π of E whose classes are taken in \mathcal{S} defines a *cut* in hierarchy H . The set of all cuts of E is denoted by $\Pi(E) = \Pi$. Every "horizontal" section $\pi_i(H)$ at level i is obviously a cut, but several levels can cooperate in a same cut, such as $\pi(S_1)$ and $\pi(S_2)$, drawn with thick dotted lines in Figure 1. Similarly, the partition $\pi(S_1) \sqcup \pi(S_2)$ generates a cut of $H(E)$. The symbol \sqcup is used here for expressing that groups of classes are concatenated. Each class S may also be the matter of cuts in the sub-hierarchy $H(S)$ whose \mathcal{S} is the summit. Let $\Pi(S)$ be the family of all cuts of $H(S)$; put

$$\tilde{\Pi} = \cup\{\Pi(S), S \in \mathcal{S}\}. \tag{3}$$

The set $\tilde{\Pi}$ does not contain all possible partial partitions with classes in \mathcal{S} . For example, the partial partition $\{a\} \sqcup \{f\}$ of Figure 1 does not belong to $\tilde{\Pi}$, though its two terms are members of \mathcal{S} . Moreover, all unions of classes are not in \mathcal{S} . If for example, S is made by union of the three hexagonal sons T_1, T_2, T_3

(Figure 2), then the hierarchy $H(S)$ may accept a cut of type 2a, but surely not of type 2b, because the union of two hexagons is not a class of \mathcal{S} . However, $\tilde{\Pi}$ contains the family $\Pi(E)$ of all cuts of $H(E)$.

The hierarchical structure of the data induces a relation between the family $\Pi(S)$ of the cuts of node S and the families $\Pi(T_1), \dots, \Pi(T_p)$ of the sons T_1, \dots, T_p of S . Since all expressions of the form $\sqcup\{\pi(T_k); 1 \leq k \leq p\}$ define cuts of S , $\Pi(S)$ contains the whole family

$$\Pi'(S) = \{\pi(T_1) \sqcup \dots \sqcup \pi(T_p); \quad \pi(T_1) \in \Pi(T_1) \dots \pi(T_p) \in \Pi(T_p)\},$$

plus the cut of S into a unique class, i.e. S itself, which is not a member of $\Pi'(S)$. And as the other unions of several T_k are not classes listed in \mathcal{S} , there is no other possible cut, hence

$$\Pi(S) = \Pi'(S) \cup S. \tag{4}$$

2.3 Properties of the Hierarchies of Partitions

The chain structure of the partitions of hierarchy H gives rise to a few nice properties about lattices, connections and ultrametrics.

Lattices. The lattice structure results from the following property 17:

Proposition 2. *The family $\Pi(E)$ of all cuts of H is a complete lattice of partitions for the ordering of refinement, where the class of the infimum of J cuts at point x is the intersection of the classes in x of the J operands, and where the supremum is their union.*

The fact that the union of the classes represents the class of the supremum derives from Proposition 13 in 13. The lattice $\Pi(E)$ is atomic. Its atoms are the partitions whose all classes but one are singletons, the last one being a pair of singletons.

Each increasing sequence $\{\pi(i), 0 \leq i \leq n\}$ of cuts whose classes come from S , and whose extremities are π_0 and π_n , defines in turn a new hierarchy H' , different from H , but of the same length, and made from classes of \mathcal{S} , just as H is. Let \mathcal{H} stands for the family of all hierarchies of this type. The refinement ordering on the partitions induces a product ordering on the families of increasing cuts, i.e. on the hierarchies of \mathcal{H} , according to which H_1 is smaller than H_2 iff at every level k the cut $\pi_1(k)$ is smaller than $\pi_2(k)$. We can state the following:

Corollary 3. *Proposition 2 shows that the set \mathcal{H} of all hierarchies having their classes in \mathcal{S} is itself a lattice. All these lattices are finite, hence complete.*

Connections. Concerning connective segmentation, the family \mathcal{S} of all classes satisfies Proposition 4 and Theorem 5 17:

Proposition 4. *Every binary criterion $\sigma : (\mathcal{F}, \mathcal{S} \cup \emptyset) \rightarrow \{0, 1\}$ is partially connective, or connective when it is satisfied by the singletons.*

In other words, every subset of \mathcal{S} containing \emptyset is a partial connection. A similar result appears in the example 21 of 14.

Ultrametrics. A distance is said to be ultrametric when the triangular inequality is replaced by the more severe following axiom [4],[3]:

$$d(x, z) \leq \max\{d(x, y), d(y, z)\}. \quad (5)$$

Two ultrametric balls can only be disjoint or concentric. Therefore equation (5) implies that the set of all balls of radius r induces a Voronoi tessellation π_r on E , whose classes are the open balls on the one hand, and the set of the frontiers on the other hand. The same equation (5) shows also that this Voronoi tessellation does not depend explicitly on a set of sites in space E , but only on radius r . When the range of variation of r is finite, then the family $H = \{\pi_r, r \geq 0\}$ forms an indexed hierarchy. Conversely, every hierarchy of partitions determines an ultrametric over the set \mathcal{S} of the classes of H [3]. Therefore, we can equivalently give ourselves an ultrametric or a hierarchy of partitions. This aspect is studied in detail by P. Arbelaez and L. Cohen in [2], by C. Ronse in [14], and by F. Meyer and L. Najman in [9].

We can try and compare the disjunctive behavior of the ultrametric with the property of the classes of a hierarchy (Equation (2)). Indeed, the three concepts of an indexed hierarchy, an ultrametric, and a universal connective criterion turn out to be three aspects of a same notion. More precisely, we have:

Theorem 5. *The three following statements are equivalent:*

1. H is an indexed hierarchy,
2. the set \mathcal{S} of all classes of the partition π_i of H forms an ultrametric space, of distance the indexing parameter,
3. every binary criterion $\sigma : (\mathcal{F}, \mathcal{S} \cup \emptyset) \rightarrow \{0, 1\}$ is partially connective.

Proof. the implication (1) \Rightarrow (3) is proved by proposition 4, and the equivalence (1) \Leftrightarrow (2) is a classical result [3]. We have to prove that (3) \Rightarrow (1). We observe firstly that, given two sets B_1 and B_2 , the criterion " $\sigma^*(f, A) = 1$ iff $A \subseteq B_1$ or $A \subseteq B_2$ " is connective iff $B_2 \subseteq B_1$ or $B_1 \subseteq B_2$. Let then H be a family of partitions. If it is not hierarchical then there exists at least one point x included in two classes B_1 and B_2 of partitions of H such that none of the two inclusions $B_2 \subseteq B_1$ and $B_1 \subseteq B_2$ is true, which implies that criterion σ^* is not connective, which achieves the proof.

This result shows that the connective approach is inefficient for hierarchies, and orients us towards the alternative method, namely the optimization of an energy.

3 Cuts of Minimum Energy in $\Pi(E)$

We set the problem as it was formulated by L.Guigues [6][5], and solved by him in the framework of additive energies and binary hierarchies. Below, the first of these two conditions is replaced by the more general relation (6) of hierarchical increasingness of the energy, and the second condition is shown to be cumbersome. In addition, the lack of unicity will lead us to analyze the structure of the solutions.

3.1 Minimization under Hierarchical Increasingness

Allocate an energy ω over the set $\tilde{\Pi}$ of partial partitions (Equation (3)), i.e. a positive numerical function $\omega : \tilde{\Pi} \rightarrow \mathbb{R}$. We propose to characterize the cut(s) of $\Pi(E) \subseteq \tilde{\Pi}$ of minimum energy. As family $\Pi(E)$ is finite, there is always at least one cut $\pi^*(E)$ of smallest energy. More generally, for each node S , the family $\Pi(S)$ of all cuts of the hierarchy of summit S admits at least one minimum cut $\pi^*(S)$.

The search for minimum cuts becomes easier when we relate the energies of the fathers, S say, to those of their sons, T say, which can be obtained by means of hierarchical increasingness:

Definition 6. *Let H be a finite hierarchy, let S be one of its nodes, and T be one of the sons of S . An energy ω on the family $\tilde{\Pi}$ of the cuts of H satisfies the condition of hierarchical increasingness in $\tilde{\Pi}$ when*

$$\omega(\pi_1(T)) \leq \omega(\pi_2(T)) \quad \Rightarrow \quad \omega[(\pi_1(T) \sqcup \pi(S \setminus T))] \leq \omega[(\pi_2(T) \sqcup \pi(S \setminus T))], \quad (6)$$

where $\pi_1(T)$ and $\pi_2(T)$ are two cuts of the sub-hierarchy of summit T , and where π is an arbitrary cut of the set difference $S \setminus T$.

The implication (6) extends an inequality relative to the partitions of set T to another inequality holding on the partitions of set S , larger than T . In that, it is a matter for partial connective segmentation, in Ch. Rouse's sense [13].

Proposition 7. *Let H be a finite hierarchy, and ω be a hierarchically increasing energy on the cuts of H . If S is a node of H with p sons $T_1..T_p$ of minimum cuts $\pi_1^*, ..\pi_p^*$, then one of the two cuts*

$$\pi_1^* \sqcup \pi_2^* .. \sqcup \pi_p^*, \quad (7)$$

or the partition of S into a unique class, is a minimum cut of S .

Proof. The hierarchical increasingness of the energy implies that cut (7) has the lowest energy among all the cuts of type $\Pi'(S) = \sqcup\{\pi(T_k); 1 \leq k \leq p\}$ (it does not follow that it is unique). Now, from the decomposition (4), every cut of S is either an element of $\Pi'(S)$, or S itself. Therefore, the set formed by the cut (7) and S contains at least one minimum cut of S .

The condition of hierarchical increasingness (6) involved in Proposition 7 is not compulsory, but it opens a broad range of energies, and is easy to check. The analysis presented in [6] and [5] focuses on the case of the separable energies, i.e. of those that satisfy the additivity relation

$$\omega(T_1 \sqcup T_2 .. \sqcup T_p) = \sum_{1 \leq k \leq p} \omega(T_k) \quad (8)$$

between the energy of a partition and those of its classes. It is the concern of a particular hierarchical increasingness. But the algorithm proposed in [6] p.142, for finding minimum cuts in case of separable energies applies indeed to any hierarchically increasing energy ω . It can be stated as follows:

Proposition 8. *Guigues' algorithm:*

- scan in one pass all nodes of H according to an ascending lexicographic order ;

- determine at each node S a temporary minimum cut of H by comparing the energy of S to that of the concatenation of the temporary minimum cuts of the (already scanned) sons T_k of S .

Remark that from Proposition 7 the temporary minimum holds always on two terms only, at each comparison, whatever the number of sons of S .

3.2 Lattice of the Minimum Cuts

The previous results never impose unicity for the minimum cuts. Each node S involves its own series of comparisons, so that it can perfectly happen that in the family $\Pi(S)$ of Relation (4) a minimum cut of $\Pi'(S)$ has the same energy as that of S . This event introduces two solutions which are then carried over the whole induction. And since such a doublet can occur regarding any node S of hierarchy H , the number of minimal cuts is *a priori* indefinite, and may be large. However, one can reduce it by some additional condition, the simplest one being the ordering of refinement itself. Indeed, at each node S , among the two possible solutions of Proposition 7, the partition of S into a single class is always the larger one for the refinement order. By ordering the solutions at each step, we thus structure them in a complete lattice whose cardinal increases according to thickness of quantification of the energy.

What advantage can we draw from this lattice structure? First of all, it solves the unicity problem. The question "find the cut that minimizes the energy" is replaced by "find the largest cut that minimizes the energy". This largest solution is characterized by the following proposition:

Proposition 9. *Let $\{x\}$ be an element of the base of a hierarchy H with $n + 1$ levels, and let $S_1(x), S_2(x), \dots, S_n(x)$ be the sequence of the nodes anterior to $\{x\}$. Let ω be hierarchically increasing energy for the hierarchy on $\Pi(E)$, and $\pi^*(E)$ be a minimum cut $H(E)$ for this energy. The class of $\pi^*(E)$ that contains $\{x\}$ is then $S_i(x)$ such that*

$$\omega[\pi^*(S_{i-1}(x))] \geq \omega[\pi^*(S_i(x))] \quad (9)$$

$$i \leq j < p \Rightarrow \omega[\pi^*(S_j(x))] < \omega[\pi^*(S_{j+1}(x))] \quad (10)$$

Instead of using the refinement, we can, alternatively, introduce a second optimization. For example, for color images, ω can hold on the luminance, and the criterion for choosing between the optimal cuts can derive from the product saturation \times hue.

3.3 Minimization with External Partitions

It may be sometimes advantageous to complete family $\widetilde{\Pi}$ of Relation (3) by additional partitions. For example, one can wish to suppress parasite small classes

by clustering them with their neighbor. This leads to introduce a set $\Xi(S)$ of partial partitions of S whose all classes are unions of more than one T_k , such as the class $\xi(T_2, T_3)$ made of the union of two hexagons in figure 2b. These supplementary cuts of S generate the following family (up to a permutation of the indexes)

$$II''(S) = \{\pi(T_1) \sqcup \dots \pi(T_k) \sqcup \xi(T_{k+1}..T_p) \quad 1 \leq k \leq p; \xi(T_{k+1}..T_p) \in \Xi(S)\}. \tag{11}$$

Family (11) has to be added to that, $II(S)$, of the legitimate sons (Relation (4)). Proposition 7 admits now the new formulation (17)

Proposition 10. *Let S be a node in hierarchy H , of p sons $T_1..T_p$ associated with p minimum cuts π_1^*, \dots, π_p^* , for some increasing energy. The minimum cuts of the set formed by*

i) the cut

$$\pi_1^* \sqcup \pi_2^* \dots \sqcup \pi_p^*, \tag{12}$$

ii) the family

$$\{\pi_1^* \sqcup \pi_2^* \dots \sqcup \pi_k^* \sqcup \xi(T_{k+1}, ..T_p) ; \xi \in \Xi(S)\}, \tag{13}$$

contains the minimum cuts of S .

Proposition 10 reduces the number of comparisons to do for finding the minimum cut of $H(S)$ from those of the sons of S . To illustrate that, we can calculate how many comparisons are necessary when S has 2, 3 or 4 sons. For two sons T_1 and T_2 , it suffices to compare the energies of the two terms $\pi_1^* \sqcup \pi_2^*$ and $S = T_1 \cup T_2$. When a third son T_3 is added, we go to five terms $\pi_1^* \sqcup \pi_2^* \sqcup \pi_3^*$, $T_1 \cup T_2 \cup T_3$, plus the three cuts of type $\pi_1^* \sqcup \xi(T_2, T_3)$ (see Figure 2). For the nodes with four sons one finds, similarly, $1 + C_4^2 + C_4^3 + 4 = 15$ possibilities. Note that all these values can only reduce when we require also that the classes of partitions ξ must be connected. Finally, the previous algorithm in one pass remains valid, but more comparisons must be performed at each node. But now the solutions no longer form a lattice, which is an obvious drawback.

4 Examples of Hierarchical Minimizations

We go back over the studies quoted in introduction, and try and interpret them in the framework that has just been developed. These studies are obviously richer than their specifics aspects of hierarchical optimization on which we concentrate here.

4.1 Suprema of Energies

We firstly contemplate energies whose law of composition is similar to the additivity relation (8), but written for the supremum. If $T_1, ..T_k, ..T_p$, stand for the classes of partition π , we thus have

$$\omega(T_1 \sqcup T_2, .. \sqcup T_p) = \sup\{\omega(T_k), 1 \leq k \leq p\}. \tag{14}$$

The energies involved in the law (14) increase hierarchically, hence lend themselves to the above optimizations. Here are two examples, where we suppose also that the energy is an increasing function of the class, i.e.

$$T \subseteq S \Rightarrow \omega(T) \leq \omega(S) \quad T, S \in \mathcal{S}. \tag{15}$$

This last condition may seem to contradict the hierarchical rise, since the partition into singletons, from which we start, becomes a solution. But it makes sense when we quantify the energy in 0 and 1, if we look for the largest element in the lattice of all solutions.

Lasso. This algorithm, due to F. Zanoguera et Al. [20], appears also in [9]. The initial image is first segmented into α -flat zones, which generate a hierarchy as the slope α increases. But the optimization itself applies to any hierarchy of segmentations. It consists in drawing manually a closed curve around the object to segment. If A designates the inside of this lasso, then we take the following function

$$\begin{aligned} \omega(S) &= 0 && \text{when } S \subseteq A ; S \in \mathcal{S} \\ \omega(S) &= 1 && \text{when not,} \end{aligned}$$

for energy, and we go from classes to partitions by the law (14) of \vee -composition of the energies. The largest cut that minimizes ω is depicted in Figure 3c. We see that the resulting contour follows the edges of the petals. Indeed, a segmented class can jump over this high gradient only for α large enough, and then this class is rejected because it spreads out beyond the limit of the lasso.

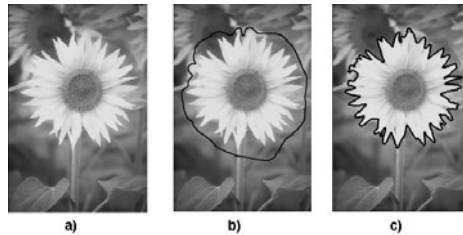


Fig. 3. a) Initial image; b) manual lasso; c) contour of the union of the classes inside the lasso

Segmentation under constraint. This method was proposed by P. Soille and J. Grazzini in [18] and [19] with several variants; it is re-formulated by C. Ronse in a more general framework in [15]. We now take for energy ω of the classes the function

$$\begin{aligned} \omega(S) &= 0 && \text{when } \sup\{f(x), x \in S\} - \inf\{f(x), x \in S\} \leq \omega_0 \\ \omega(S) &= 1 && \text{when not,} \end{aligned}$$

where ω_0 is a given threshold, and we go to partitions by \vee -composition. In the examples of [18] and [19], the hierarchy is obtained by α -flat zones, and,

for multi-spectral images, by the intersection of the α -flat zones of the various bands. Sometimes the criterion "there is at least one extremum of f inside S " is added. The class at point x of the largest partition of minimum energy is given by the largest $S \in \mathcal{S}$, that contains x , and such that the amplitude of variation of f inside S be $\leq \omega_0$.

4.2 Additive Energies

Separable models. They are energies whose archetype is given by the classical Mumford and Shah functional. One constructs a separable energy in \mathbb{R}^2 or in \mathbb{Z}^2 by starting from a partition of S into its sons T_1, \dots, T_p plus the set Γ of their frontiers. A first measure ω_μ holds on the T_k , and a second one, ω_ν , on their frontiers ∂T_k , and we put

$$\omega(S) = \sum_{1 \leq k \leq p} \omega_\mu(T_k) + \omega_\nu(\partial T_k), \quad (16)$$

(according to Integral Geometry, a third term, proportional to Euler-Poincaré Constant could be added). This models are said to be separable [6] [5] in that we have $\omega(S) = \omega(T_1) + \dots + \omega(T_p)$, up to external edge effects. The energy ω , additive, is thus hierarchically increasing. Different instructive variants may be found in [6].

Thumbnails. The creation of digital thumbnails by Ph. Salembier and L. Garrido [16] is a matter for separable models. One aims to generate reduced versions of a given color image f from a hierarchy of its segmentations. In each class T function f is replaced by its mean value $m(T)$. The approximation is estimated in the L_2 norm, i.e.

$$\omega_\mu(T) = \sum_{x \in R} \| f(x) - m(T) \|^2.$$

If the coding cost for a frontier element is k , that of the whole class T becomes

$$\omega_\nu(T) = 24 + \frac{k}{2} |\partial T|$$

with 24 bits for $m(T)$. The total energy of a cut is written $\omega(S) = \lambda \omega_\mu(S) + \omega_\nu(S)$. For a fixed λ , it yields a first minimization, followed by a second one relatively to the Lagrange parameter λ .

4.3 Other Laws of Composition

A number of laws of composition are compatible with hierarchically increasing energies. Instead of supremum and sum, that we just presented, one could use infimum, product, difference sup-inf, quadratic sum, etc., and all their combinations. Moreover, one can make depend ω on more than one class, on the proximity of the edges, on several elements of the lattice \mathcal{H} of the hierarchies, etc...

Here is an example of another law of composition, which extends the technique developed in [1]. Start from three partition hierarchies H_l , H_s , and H_h of luminance, saturation, and hue of a given color image. The purpose is to reduce them to a unique significant cut, and the idea is that the higher (resp. lower) the saturation, the more representative is the hue (resp. luminance). Therefore, when a region has a weak (resp. strong) saturation, then its luminance (hue) is priority for the segmentation. Suppose that saturation s varies between 0 and 1. Provide the classes of H_s with the energy

$$\omega(T_s) = \frac{1}{\text{area } T_s} \left[1 - \int_{T_s} s(x) dx \right], \quad (17)$$

and weight by areas for expressing the energies of the partitions from those of the classes. It results in the minimum cut $\bar{\pi}_s$. Repeat the process by replacing $1 - \int_{T_s} s(x) dx$ by $\int_{T_s} s(x) dx$ in (17), which results in the new energy $\omega'(T_s)$ and the new minimum cut $\bar{\pi}'_s$. Intersect then the two partitions $\bar{\pi}_s$ and $\bar{\pi}'_s$. When the class at point x of the intersection has an energy $\omega - \omega' > 0$, it is labelled "class for luminance", and when not, "class for hue". The process ends by taking the partial hierarchy of the luminance (resp. the hue) in the union of the classes for luminance (resp. for hue), and segmenting luminance and hue individually, in their own domains.

5 Conclusion

Considering the hierarchies of partitions of an image as a segmentation tool, we looked for the meaning of an optimum cut, for laws able to govern this optimum. We showed that the assumption of hierarchical increasingness (6) allows to regroup various segmentation techniques which seem to be far away from each other, and we proposed new ones.

Acknowledgement. The author wishes to thank *J. Angulo, J. Cousty, L. Najman, and Ch. Ronse* for their valuable comments.

References

1. Angulo, J., Serra, J.: Modeling and segmentation of colour images in polar representations. *Image and Vision Computing* 25, 475–495 (2007)
2. Arbelaez, P., Cohen, L.D.: Segmentation d'images couleur par partitions de Voronoï. *Traitement du Signal* 21(5), 407–421 (2004)
3. Benzécri, J.P.: *L'Analyse des Données. Tome I: La Taxinomie*, Dunod, Paris, 4e édition (1984)
4. Diday, E., Lemaire, J., Pouget, J., Testu, F.: *Eléments d'analyse des données*, Dunod, Paris (1982)
5. Guigues, L., Cocquerez, J.P., Le Men, H.: Scale-Sets Image Analysis. *Int. Journal of Computer Vision* 68(3), 289–317 (2006)
6. Guigues, L.: *Modèles multi-échelles pour la segmentation d'images*. Thèse doctorale Université de Cergy-Pontoise, décembre (2003)

7. Marr, D.: *Vision*. Freeman and Co., New York (1982)
8. Matheron, G.: *Random Sets and Integral Geometry*. Wiley, New-York (1975)
9. Meyer, F., Najman, L.: *Segmentation, Minimum Spanning Trees, and Hierarchies*. In: Najman, L., Talbot (eds.) *Mathematical Morphology*. Wiley, N.Y. (2010)
10. Najman, L.: *Ultrametric watersheds*. In: Wilkinson, M.H.F., Roerdink, B.T.M. (eds.) *ISMM 2009*. LNCS, vol. 5720, pp. 181–192. Springer, Heidelberg (2009)
11. Najman, L.: *Ultrametric watersheds: a bijection theorem for hierarchical edge-segmentation*. In: *JMIV* (to appear, 2011)
12. Noyel, G., Angulo, J., Jeulin, D.: *On distances, paths and connections for hyper-spectral image segmentation*. In: *Proceedings of the 8th International Symposium on Mathematical Morphology*, Rio de Janeiro, Brazil, October 10-13. MCT/INPE, vol. 1, pp. 399–410 (2007)
13. Ronse, C.: *Partial partitions, partial connections and connective segmentation*. *Journal of Mathematical Imaging and Vision* 32, 97–125 (2008)
14. Ronse, C.: *Adjunctions on the lattices of partitions and of partial partitions*. *AAECC* 21(5), 343–396 (2010)
15. Ronse, C.: *Idempotent block splitting on partial partitions, I: isotone operators, II: non-isotone operators* *Order* (to appear, 2011)
16. Salembier, P., Garrido, L.: *Binary Partition Tree as an Efficient Representation for Image Processing, Segmentation, and Information Retrieval*. *IEEE Trans. on Image Processing* 9(4), 561–576 (2000)
17. Serra, J.: *Ordre de la construction et segmentations hié rarchiques*. Colloque ESIEE (April 2, 2010), <http://laurentnajman.org/serra70/index>
18. Soille, P.: *Constrained connectivity for hierarchical image partitioning and simplification*. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 1132–1145 (2008)
19. Soille, P., Grazzini, J.: *Constrained Connectivity and Transition Regions*. In: Wilkinson, M.H.F., Roerdink, B.T.M. (eds.) *Mathematical Morphology and its Applications to Signal and Image Processing*, pp. 59–70. Springer, Heidelberg (2009)
20. Zanoguera, F., Marcotegui, B., Meyer, F.: *A toolbox for interactive segmentation based on nested partitions*. In: *Proc. of ICIP 1999*, Kobe, Japan (1999)

An Arithmetic and Combinatorial Approach to Three-Dimensional Discrete Lines

Valérie Berthé¹ and Sébastien Labbé^{2,*}

¹ Laboratoire d'Informatique Algorithmique : Fondements et Applications,
Université Paris Diderot, Paris 7 - Case 7014
F-75205 Paris Cedex 13, France

`berthe@liafa.jussieu.fr`

² Laboratoire de Combinatoire et d'Informatique Mathématique,
Université du Québec à Montréal,
C.P. 8888 Succursale "Centre-Ville", Montréal (QC), Canada H3C 3P8
`labbe.sebastien@courrier.uqam.ca`

Abstract. The aim of this paper is to discuss from an arithmetic and combinatorial viewpoint a simple algorithmic method of generation of discrete segments in the three-dimensional space. We consider discrete segments that connect the origin to a given point (u_1, u_2, u_3) with co-prime nonnegative integer coordinates. This generation method is based on generalized three-dimensional Euclid's algorithms acting on the triple (u_1, u_2, u_3) . We associate with the steps of the algorithm substitutions, that is, rules that replace letters by words, which allow us to generate the Freeman coding of a discrete segment. We introduce a dual viewpoint on these objects in order to measure the quality of approximation of these discrete segments with respect to the corresponding Euclidean segment. This viewpoint allows us to relate our discrete segments to finite patches that generate arithmetic discrete planes in a periodic way.

Keywords: Discrete Segments, Discrete Lines, Christoffel words, multi-dimensional Euclid's algorithms, multi-dimensional continued fractions, substitutions.

1 Introduction

Discrete lines and segments in the plane are quite well understood and their study has already arised a vast literature on the subject (see e.g. the references in [14]). The Freeman codings of arithmetic standard discrete lines correspond to the also well-studied family of Sturmian words. For more details, see e.g. [15]. Among the factors of Sturmian words, Christoffel words play a particular role and correspond to Freeman codings of segments. The deep and fruitful connections between Sturmian words and continued fractions, on the one hand, and between Christoffel words and Euclid's algorithm, on the other hand, allows a thorough description of most of their properties (see Figure 1 for an illustration).

* With support of the NSERC. The authors also would like to thank P. Arnoux and X. Provençal for many fruitful discussions, and the referees for their valuable remarks.

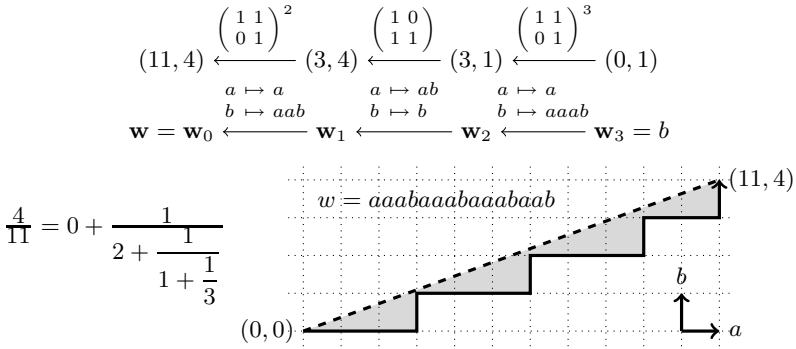


Fig. 1. It is well known that the reduction of a two-dimensional integer vector by using Euclid’s algorithm allows one to construct the discrete segment (also called Christoffel word) which is such that no integer point is in the interior of the gray region. Its Freeman coding $\mathbf{w} = aaabaaabaabaab$ can be obtained by applying on the letter b the substitutions associated with the steps of Euclid’s algorithm performed on $(11, 4)$.

There exist various strategies for defining and generating discrete lines in the three-dimensional space. With no claim for being exhaustive, let us quote e.g. [11,12,13,8,9]. The approach we follow here is motivated by the study of Sturmian words. Several generalizations of Sturmian words over a three-letter alphabet have been considered. For instance, infinite words coding trajectories in a cubic billiard have been investigated in [3]. One of their drawbacks is that they produce infinite words having a quadratic number of factors of a given length, which seems to indicate that there is no suitable continued fraction algorithm allowing one to describe them. This prevents in particular multiscale analysis. Another direction of generalization of Sturmian words consists in working with balanced words over a three-letter alphabet. However balanced words over a higher-alphabet do not seem to be good candidates for describing discrete segments in the space, as shown in [13]. The family of Arnoux-Rauzy words [4] provides a third fruitful way of generalizing Sturmian words. They have a linear number of factors of a given length ($2n + 1$ factors of length n), and can be described in terms of a multi-dimensional continued fraction algorithm. Nevertheless this algorithm is not defined everywhere, and thus, they cannot be used to approximate all the slopes in the space. For more details, see the discussion and the references in [5].

Our strategy here works in the reverse direction: we start from Euclid’s algorithms that are defined everywhere and we associate with them families of words. More precisely, we want to construct discrete segments that connect the origin to a given point (u_1, u_2, u_3) with coprime nonnegative integer coordinates. We apply a three-dimensional Euclid’s algorithm to the triple (u_1, u_2, u_3) (see Section 2.1). In Section 2.2 we associate with the steps of the algorithm substitutions, that is, rules that replace letters by words, which allow us to generate the Freeman coding of the discrete segment.

We thus obtain in Section 3 a simple algorithmic way for producing discrete segments and lines by means of substitutions and generalized Euclid's algorithms which can allow a multiscale approach for their study. Our description is both analytic and arithmetic. Note that by discrete segment, we mean here broken lines constructed by concatenating unit steps oriented along the three coordinate axes. Based on the quality of approximation of the underlying multi-dimensional continued fraction algorithms that are used (see Remark 2 below), we expect these segments to be good candidates for discretizations of Euclidean segments and lines, which is supported by the experimental studies we conducted.

Section 4 aims at getting a theoretical and dynamical understanding of the way these segments approximate Euclidean segments by introducing a discrete plane that is transverse to the original direction (u_1, u_2, u_3) . The present paper relies on a formalism that has been previously introduced in a different context in 2. The new notions that are introduced here correspond mainly to Definition 3 and to the choice of the normal vector \mathbf{v} of the transverse plane in Equation (1).

2 Preliminaries

Let $\mathbf{u} = (u_1, u_2, u_3)$ be a vector with coprime entries in \mathbb{N}^3 . We want to introduce a discrete line approximating the vectorial line directed by the vector \mathbf{u} in \mathbb{R}^3 .

2.1 Generalized Euclid's Algorithm

In the one-dimensional case, most of the existing continued fraction algorithms strongly rely on Euclid's algorithm: starting from two nonnegative numbers a and b , one subtracts the smallest one to the largest one. If one performs only one subtraction at each step, one obtains the so-called *additive* version of Euclid's algorithm. If one performs in one step as much subtractions as possible (i.e., if $0 \leq b \leq a$, a is replaced by $a - [a/b]b$), one gets a *multiplicative* algorithm. In the multi-dimensional case, there is no such canonical algorithm, and several different definitions have been proposed (see 17 for a summary). Indeed starting from more than two numbers, it is not clear to decide which operation is to be performed on these numbers, hence the diversity of existing generalizations of Euclid's algorithm (see Section 2.3).

We will thus use the following framework for defining versions of three-dimensional generalizations of Euclid's algorithms. Let \mathcal{M}_E be the set of 3×3 matrices $M = [m_{ij}]_{1 \leq i, j \leq 3}$ with entries in $\{0, 1\}$ having only 1's on the diagonal and exactly one nonzero entry m_{ij} with $i \neq j$. Let \mathcal{M}_P be the set of 3×3 matrices that are *permutation matrices*, that is, they have entries in $\{0, 1\}$, and only one nonzero coefficient on each line and on each column. Matrices of \mathcal{M}_E and of \mathcal{M}_P are called *elementary matrices*. We set \mathcal{M} to be the set of finite products of matrices of $\mathcal{M}_E \cup \mathcal{M}_P$.

Definition 1 (Three-dimensional Euclid's algorithm). *Let*

$$X \subset \{(u_1, u_2, u_3) \mid \forall i, u_i \in \mathbb{N}, \text{ and } \gcd(u_1, u_2, u_3) = 1\}$$

and let $X_f \subset X$. Elements of X_f are called terminal.

A three-dimensional Euclid's algorithm is a map $T : X \rightarrow X$ such that $T(x) = x$ for all $x \in X_f$, for any $\mathbf{u} \in X$ there is $M \in \mathcal{M}$ satisfying $\mathbf{u} = MT(\mathbf{u})$, and for every $\mathbf{u} \in X$ there exists N such that $T^N(\mathbf{u}) \in X_f$.

2.2 Euclid's Substitutions

Let us consider a finite set of letters \mathcal{A} called *alphabet*. A (finite) *word* is an element of the free monoid \mathcal{A}^* generated by \mathcal{A} . A *substitution* σ over the alphabet \mathcal{A} is an endomorphism of the free monoid \mathcal{A}^* . It is completely defined by its image on the letters of the alphabet. For $i \in \{1, 2, 3\}$ and for $w \in \{1, 2, 3\}^*$, let $|w|_i$ stand for the number of occurrences of the letter i in the word w . The map

$$l : \{1, 2, 3\}^* \rightarrow \mathbb{N}^n, \quad w \mapsto {}^t(|w|_1, |w|_2, |w|_3)$$

is called the *abelianization map*. Notice that in the literature, this map is also referred to as the *Parikh mapping*. Let σ be a substitution on $\{1, 2, 3\}^*$. Its *incidence matrix* or *abelianized matrix* $M_\sigma = (m_{i,j})_{1 \leq i, j \leq 3}$ is defined as the square matrix with entries $m_{i,j} = |\sigma(j)|_i$ for all i, j . We say that σ is *unimodular* if $\det(M_\sigma) = \pm 1$.

Definition 2 (Three-dimensional Euclid's substitutions). Let T be a three-dimensional Euclid's algorithm. With each matrix $M \in \mathcal{M}$ produced by the algorithm, we associate a substitution whose incidence matrix is given by M .

Remark 1. Given a matrix produced by a Euclid's algorithm, there exist several substitutions having this matrix as incidence matrix. The substitutions generating words that are Freeman codings of discrete segments are known to be Sturmian. Given an incidence matrix $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ of a Sturmian substitution, only $a+b+c+d-1$ substitutions having this matrix as incidence matrix are Sturmian (i.e., preserve discrete segments). For more details, see [15] and the references therein. Hence, the choice of a substitution associated with an incidence matrix can play an important role. This is why we try to privilege as much as possible here additive steps. We thus usually recover elementary matrices or simple products of them, which reduces the choices for the associated substitution.

Example 1. With the elementary matrix $M = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ are associated

$$\sigma : 1 \mapsto 12, \quad 2 \mapsto 2, \quad 3 \mapsto 3 \quad \text{and} \quad \tilde{\sigma} : 1 \mapsto 21, \quad 2 \mapsto 2, \quad 3 \mapsto 3.$$

2.3 A Zoo of Algorithms

We recall here the most classical generalizations of Euclid's algorithms which have lead to well-studied multi-dimensional continued fraction algorithms such as those discussed in [17]:

- Jacobi-Perron: let $0 \leq u_1, u_2 \leq u_3$

$$(u_1, u_2, u_3) \mapsto (u_2 - [\frac{u_2}{u_1}]u_1, u_3 - [\frac{u_3}{u_1}]u_1, u_1),$$

- Brun: we subtract the second largest entry to the largest one; for instance, if $0 \leq u_1 \leq u_2 \leq u_3$,

$$(u_1, u_2, u_3) \mapsto (u_1, u_2, u_3 - u_2);$$

- Poincaré: we subtract the second largest entry to the largest one, and the smallest entry to the second largest one; for instance, if $0 \leq u_1 \leq u_2 \leq u_3$

$$(u_1, u_2, u_3) \mapsto (u_1, u_2 - u_1, u_3 - u_2),$$

- Selmer: we subtract the smallest positive entry to the largest one; for instance, if $0 < u_1 \leq u_2 \leq u_3$

$$(u_1, u_2, u_3) \mapsto (u_1, u_2, u_3 - u_1),$$

- Fully subtractive: we subtract the smallest positive entry to all the largest ones; for instance, if $0 < u_1 \leq u_2 \leq u_3$

$$(u_1, u_2, u_3) \mapsto (u_1, u_2 - u_1, u_3 - u_1).$$

We have recalled here Jacobi-Perron algorithm in its multiplicative form for the sake of clarity, but an additive version of this algorithm can be given. Furthermore, one checks that we can chose as terminal set for all these algorithms the set

$$X_f = \{(1, 0, 0), (0, 1, 0), (0, 0, 1)\} \subset X,$$

by possibly applying the two-dimensional Euclid's algorithm once the first coordinate has reached the value 0 in the Jacobi-Perron case.

Remark 2. The choice of these algorithms is motivated by the quality of approximation they provide. Indeed, Jacobi-Perron and Brun algorithm are known to provide almost everywhere exponential convergence (see [10]).

Example 2. Let $\mathbf{u} = (2, 2, 3)$. By using Brun algorithm, one has $\mathbf{u}_0 = (2, 2, 3)$, $\mathbf{u}_1 = (2, 2, 1)$, $\mathbf{u}_2 = (0, 2, 1)$, $\mathbf{u}_3 = (0, 1, 1)$, $\mathbf{u}_4 = (0, 0, 1)$. By using Poincaré algorithm, one obtains $\mathbf{u}_0 = (2, 2, 3)$, $\mathbf{u}_1 = (2, 0, 1)$, $\mathbf{u}_2 = (1, 0, 1)$, $\mathbf{u}_3 = (1, 0, 0)$.

3 A Generation Method for Discrete Segments

Let us apply to \mathbf{u} a finite sequence of steps under the action of one of the three-dimensional Euclid's algorithm T given in Section 2.3 with X_f defined as above together with a choice of Euclid's substitutions associated with the produced matrices. One has $\mathbf{u} = M_1 \cdots M_N \mathbf{u}_N$, where the vector $\mathbf{u}_N \in X_f$ has only two coordinates equal to 0, and one coordinate equal to 1. Let $\mathbf{w}_N \in \{1, 2, 3\}$ be the unique word (of length one) such that $l(\mathbf{w}_N) = \mathbf{u}_N$. The associated Euclid's substitutions are denoted by σ_n , for $1 \leq n \leq N$ (see the diagram below).

$$\mathbf{u} = \mathbf{u}_0 \xrightarrow{M_1^{-1}} \mathbf{u}_1 \xrightarrow{M_2^{-1}} \mathbf{u}_2 \xrightarrow{M_3^{-1}} \dots \xrightarrow{M_N^{-1}} \mathbf{u}_N \in X_f$$

$$\mathbf{w} = \mathbf{w}_0 \xleftarrow{\sigma_1} \mathbf{w}_1 \xleftarrow{\sigma_2} \mathbf{w}_2 \xleftarrow{\sigma_3} \dots \xleftarrow{\sigma_N} \mathbf{w}_N \in \{1, 2, 3\}$$

Definition 3 (Discrete segment). *The discrete segment associated with the vector \mathbf{u} and with the three-dimensional Euclid’s algorithm T is defined as the broken line with integer vertices that starts from the origin, whose Freeman coding is given by the coding word*

$$\mathbf{w} := \sigma_1 \cdots \sigma_N(\mathbf{w}_N).$$

In other words, the vertices of this broken line are given by the abelianized by \mathbf{l} of the prefixes of the word \mathbf{w} .

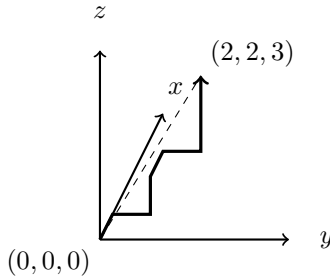
Example 3. If $\mathbf{u} = (2, 2, 3)$ and by using Poincaré’s algorithm, one has $\mathbf{w}_N = \mathbf{w}_3 = 1$ and $\mathbf{w} = \mathbf{w}_0 = 1231233$.

$$(2, 2, 3) \xrightarrow{\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}^{-1}} (2, 0, 1) \xrightarrow{\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}^{-1}} (1, 0, 1) \xrightarrow{\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}^{-1}} (1, 0, 0)$$

1 ↦ 123	1 ↦ 1	1 ↦ 13
2 ↦ 23	2 ↦ 123	2 ↦ 123
3 ↦ 3	3 ↦ 13	3 ↦ 3

$$\mathbf{w}_0 \xleftarrow{\quad} \mathbf{w}_1 \xleftarrow{\quad} \mathbf{w}_2 \xleftarrow{\quad} \mathbf{w}_3$$

The discrete segment is depicted below. Its Euclidean distance to the Euclidean segment is 1.3720.



4 A Dual Viewpoint

In this section, we introduce the following notation: one sets $M_{i..j} := M_i \cdots M_j$ and $\sigma_{i..j} := \sigma_i \cdots \sigma_j$ for $1 \leq i, j \leq N$. Hence, the incidence matrix of the substitution $\sigma_{i..j}$ is $M_{i..j}$.

In order to study the quality of approximation of the vector line directed by \mathbf{u} provided by the discrete segment \mathbf{w} , we introduce a transverse plane that does

not contain vector \mathbf{u} . Such a plane can be described by its normal vector \mathbf{v} that we chose with positive entries and not collinear with \mathbf{u} . The vector having all entries equal to 1 is denoted by $\mathbf{1}$. We chose for \mathbf{v}

$$\mathbf{v} := {}^t M_{1..N} \cdot \mathbf{1} = {}^t M_N \cdots {}^t M_1 \cdot \mathbf{1}. \tag{1}$$

We furthermore write $\mathbf{w} = z_1 \cdots z_k \cdots z_{|\mathbf{w}|}$ where $z_k \in \{1, 2, 3\}$ are letters. The vertices of the discrete segment are thus of the form $p_k = \mathbf{l}(z_1 \cdots z_k)$, for $1 \leq k \leq |\mathbf{w}|$. The choice of vector \mathbf{v} is motivated by the following relation that we will use below

$$\begin{aligned} \langle p_k, \mathbf{1} \rangle &= \langle (M_{1..N})^{-1} \cdot p_k, {}^t M_{1..N} \cdot \mathbf{1} \rangle \\ &= \langle (M_{1..N})^{-1} \cdot p_k, \mathbf{v} \rangle. \end{aligned} \tag{2}$$

The aim of this section is to relate vertices of the discrete segment to faces of a finite pattern of the discrete plane with normal vector \mathbf{v} via the mapping $(M_{1..N})^{-1}$, and to interpret the coding word \mathbf{w} in terms of a coding of the orbit of a point under a dynamical system acting on this discrete plane with normal vector \mathbf{v} . For that purpose, we introduce in Section 4.2 a dual notion of substitution acting on faces of discrete planes.

4.1 Discrete Planes

Let \mathbf{n} be a nonzero vector in \mathbb{N}^3 . According to [16], we recall that the arithmetic standard plane $\mathfrak{P}_{\mathbf{n}}$ of normal vector $\mathbf{n} = (n_1, n_2, n_3)$ is defined as

$$\mathfrak{P}_{\mathbf{n}} = \{ \mathbf{x} \in \mathbb{Z}^3 \mid 0 < \langle \mathbf{x}, \mathbf{n} \rangle \leq \|\mathbf{n}\|_1 = n_1 + n_2 + n_3 \}.$$

For $\mathbf{x} \in \mathbb{Z}^3$ and $i \in \{1, 2, 3\}$, let (\mathbf{x}, i^*) stand for the pointed face defined as the translation by \mathbf{x} of the surfel generated by $\{e_1, e_2, e_3\} \setminus \{e_i\}$ (see Figure 2). We say that \mathbf{x} is the *vertex* and i is the *type* of the pointed face (\mathbf{x}, i^*) .

$$\begin{aligned} (\mathbf{x}, 1^*) &:= \mathbf{x} + \{ \lambda e_2 + \mu e_3, (\lambda, \mu) \in [0, 1]^2 \} \\ (\mathbf{x}, 2^*) &:= \mathbf{x} + \{ \lambda e_1 + \mu e_3, (\lambda, \mu) \in [0, 1]^2 \} \\ (\mathbf{x}, 3^*) &:= \mathbf{x} + \{ \lambda e_1 + \mu e_2, (\lambda, \mu) \in [0, 1]^2 \}. \end{aligned}$$

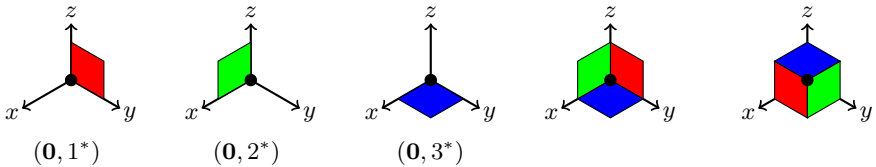


Fig. 2. Left: Geometric interpretation of faces. Right: Lower unit cube and upper unit cube. In this figure and the following, the vertex $(0, 0, 0)$ or $(1, 1, 1)$ is identified by a black dot.

We will use the following notation for translates of faces: if (\mathbf{x}, i^*) is a face and \mathbf{y} is a vector, then $(\mathbf{x}, i^*) + \mathbf{y} := (\mathbf{x} + \mathbf{y}, i^*)$ which extends in a natural way to union of faces. The lower unit cube refers to the set $\{(\mathbf{0}, 1^*), (\mathbf{0}, 2^*), (\mathbf{0}, 3^*)\}$, whereas the upper unit cube refers to $\{(\mathbf{e}_1, 1^*), (\mathbf{e}_2, 2^*), (\mathbf{e}_3, 3^*)\}$ (see Figure 2).

Let $\mathcal{P}_{\mathbf{n}}$ be the set of pointed faces satisfying

$$\mathcal{P}_{\mathbf{n}} = \{(\mathbf{x}, i^*) \mid 0 \leq \langle \mathbf{x}, \mathbf{n} \rangle < n_i\}. \tag{3}$$

One checks that the points of $\mathfrak{P}_{\mathbf{n}}$ are the vertices (*i.e.*, the corners) of the faces of $\mathcal{P}_{\mathbf{n}}$. By abuse of terminology, by arithmetic discrete plane with normal vector \mathbf{n} , we mean in all that follows this union of pointed faces $\mathcal{P}_{\mathbf{n}}$. Note that in particular, if \mathbf{n} has positive entries, the lower unit cube is included in $\mathcal{P}_{\mathbf{n}}$.

Furthermore, for any vertex $p_k = \mathbf{l}(z_1 \cdots z_k)$ of the discrete segment, one has

$$0 \leq \langle p_k, \mathbf{1} \rangle \leq \langle \mathbf{u}, \mathbf{1} \rangle = \langle M_{1..N} \cdot \mathbf{u}_N, \mathbf{1} \rangle = \langle \mathbf{u}_N, {}^t M_{1..N} \cdot \mathbf{1} \rangle = \langle \mathbf{u}_N, \mathbf{v} \rangle = v_{\mathbf{w}_N}.$$

Hence by (2) and (3) the vertices p_k of the discrete segment are mapped by $(M_{1..N})^{-1}$ onto vertices of faces of type \mathbf{w}_N of the discrete plane $\mathcal{P}_{\mathbf{v}}$. The aim of the next section is to investigate this relation.

4.2 Generalized Substitutions

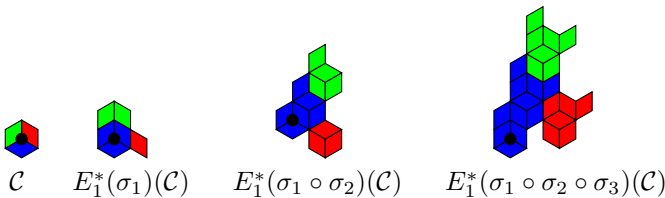
When σ is a unimodular substitution, it is possible to associate with it a notion of substitution acting on faces of cubes, following the formalism of [2]:

$$E_1^*(\sigma)(\mathbf{x}, i^*) := \sum_{j \in \{1,2,3\}} \sum_{p,s \text{ such that } \sigma(j)=pi_s} (M_{\sigma}^{-1}(\mathbf{x} + \mathbf{l}(s)), j^*). \tag{4}$$

The action of $E_1^*(\sigma)$ extends in a natural way to unions of faces. A mapping of the form $E_1^*(\sigma)$ is called a *generalized substitution*. It is obtained as the dual map of some map $E_1(\sigma)$ that can be seen as a geometric realization of σ . In the notation $E_1^*(\sigma)$, the subscript of $E_1^*(\sigma)$ stands for the codimension of the faces, while the superscript of $E_1^*(\sigma)$ refers to duality. Note that the incidence matrix of $E_1^*(\sigma)$ is the transpose of the incidence matrix of σ .

Example 4. Let $\sigma_1 : 1 \mapsto 123, 2 \mapsto 23, 3 \mapsto 3$, $\sigma_2 : 1 \mapsto 1, 2 \mapsto 123, 3 \mapsto 3$, $\sigma_3 : 1 \mapsto 13, 2 \mapsto 123, 3 \mapsto 3$ be the substitutions obtained from the reduction of the vector $(2, 2, 3)$ by using Poincaré algorithm.

If $\mathcal{C} = \text{cube}$ is the lower unit cube, then one gets



One key property of generalized substitutions is that they preserve discrete planes, as proved in [2]. Indeed, one has the following result for any unimodular substitution σ and any vector \mathbf{n} with nonnegative entries:

$$E_1^*(\sigma)(\mathcal{P}_{\mathbf{n}}) = \mathcal{P}_{t_{M_\sigma}\mathbf{n}}. \tag{5}$$

4.3 Dual Pattern

We can now apply the notions previously introduced in order to define a pattern of the discrete plane $\mathcal{P}_{\mathbf{v}}$ (see (II)) that can be associated with the coding word \mathbf{w} .

Definition 4 (Dual pattern). For $i \in \{1, 2, 3\}$, let

$$\mathcal{W}_i := E_1^*(\sigma_N) \circ \dots \circ E_1^*(\sigma_1)(\mathbf{0}, i^*)$$

and

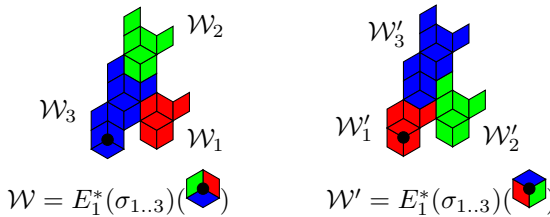
$$\mathcal{W}'_i := E_1^*(\sigma_N) \circ \dots \circ E_1^*(\sigma_1)(\mathbf{e}_i, i^*).$$

One sets furthermore $\mathcal{W} = \mathcal{W}_1 \cup \mathcal{W}_2 \cup \mathcal{W}_3$ and $\mathcal{W}' = \mathcal{W}'_1 \cup \mathcal{W}'_2 \cup \mathcal{W}'_3$.

According to [2], the three patterns \mathcal{W}_i (resp. \mathcal{W}'_i) for $i \in \{1, 2, 3\}$ have disjoint interiors. Furthermore \mathcal{W} and \mathcal{W}' coincide except on faces of the lower and upper unit cubes, \mathcal{W} contains the lower unit cube, and \mathcal{W}' the upper one.

Remark 3. The pattern \mathcal{W} is obtained by taking the image of the lower unit cube under the action of $E_1^*(\sigma)$. Note that \mathbf{w} and \mathcal{W} do not have the same number of elements. Indeed the number of elements of the coding word \mathbf{w} is equal to the sum of entries of the column with index \mathbf{w}_N of $M_{1..N}$, whereas the number of elements of the pattern \mathcal{W}_i is equal to the sum of entries of the line with index i of $M_{1..N}$, by (4). Nevertheless, the number of faces in \mathcal{W} is equal to $\sum_{i=1}^3 |\sigma_{1..N}(i)|$.

Example 5. Let $\mathbf{u} = (2, 2, 3)$. By using Poincaré algorithm, both \mathcal{W} and \mathcal{W}' have 24 faces:



The following theorem summarizes the main properties of the dual pattern. This theorem is an adaptation to the present context of results of [2].

Theorem 1. *The following properties hold:*

1. $\mathcal{W} \subset \mathcal{P}_{\mathbf{v}}$;
2. the pattern \mathcal{W} is a periodic pattern for $\mathcal{P}_{\mathbf{v}}$ with period vectors being

$$(M_{1..N})^{-1}(\mathbf{e}_1 - \mathbf{e}_2), (M_{1..N})^{-1}(\mathbf{e}_1 - \mathbf{e}_3);$$

3. for all $i \in \{1, 2, 3\}$, one has

$$\mathcal{W} + (M_{1..N})^{-1}e_i \subset \mathcal{W}'.$$

Proof. 1. We first note that the faces $(0, i^*) \subset \mathcal{P}_1$ by (3) for $i = 1, 2, 3$. We deduce the first assertion from $\mathbf{v} = {}^t M_{1..N} \mathbf{1}$ and from (5).

2. One has for $i \neq j$ $\langle (M_{1..N})^{-1}(e_i - e_j), \mathbf{v} \rangle = \langle e_i - e_j, \mathbf{1} \rangle = 0$. Hence for every $m, n \in \mathbb{Z}$, $\mathcal{W} + m(M_{1..N})^{-1}(e_1 - e_2) + n(M_{1..N})^{-1}(e_1 - e_3) \subset \mathcal{P}_v$.

3. Let $(\mathbf{x}, k^*) \subset \mathcal{W}_i$. By definition, the face (\mathbf{x}, k^*) occurs in the image by $E_1^*(\sigma_{1..N})$ of the face $(0, i^*)$. Hence, there exists s such that $\sigma_{1..N}(k) = pis$. One has $\mathbf{x} = (M_{1..N})^{-1}\mathbf{l}(s)$.

We assume that p is not equal to the empty word. Let j stand for its last letter. The face $(\mathbf{x} + (M_{1..N})^{-1}e_i, k^*)$ occurs in the image of the face $(0, j^*)$ by $E_1^*(\sigma_{1..N})$, by considering as suffix is . Hence it occurs in \mathcal{W} and thus also in \mathcal{W}' since both sets coincide except on the lower and upper unit cubes.

Assume now that p is equal to the empty word. One has

$$\begin{aligned} \mathbf{x} &= (M_{1..N})^{-1}(\mathbf{l}(\sigma(k)) - e_i) = (M_{1..N})^{-1}(M_{1..N}e_k - e_i) \\ &= e_k - (M_{1..N})^{-1}e_i. \end{aligned}$$

Hence the face $(\mathbf{x} + (M_{1..N})^{-1}e_i, k^*) = (e_k, k^*)$ occurs in \mathcal{W}' .

4.4 Exchange of Pieces

According to [2], Theorem 1 allows one to define a mapping from \mathcal{W} onto \mathcal{W}' defined as an exchange of pieces between both sets.

Definition 5. We define the mapping

$$\mathcal{E}: \mathcal{W} \rightarrow \mathcal{W}', (\mathbf{x}, k^*) \mapsto (\mathbf{x} + (M_{1..N})^{-1}e_i, k^*) \text{ if } (\mathbf{x}, k^*) \in \mathcal{W}_i.$$

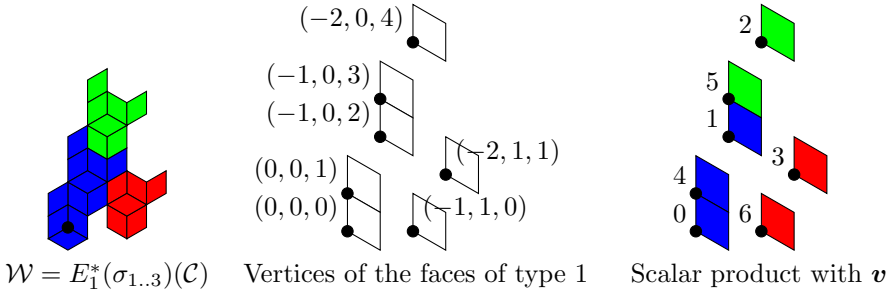
This definition is illustrated in Example 5. We have seen in the proof of the third assertion of Theorem 1 that $\mathcal{E}^k(0, i^*) \in \mathcal{W}$ for $0 \leq k < |\sigma_{1..N}(i)|$ and that $\mathcal{E}^k(0, i^*) = (e_i, i^*)$ for $k = |\sigma_{1..N}(i)|$. We define the coding of the orbit of $(0, i^*)$ under the action of \mathcal{E} as the word of length $|\sigma_{1..N}(i)|$ defined over the alphabet $\{1, 2, 3\}^*$ as follows: for $1 \leq k \leq |\sigma_{1..N}(i)|$, its k th letter is equal to the index j of the subpattern \mathcal{W}_j to which $\mathcal{E}^{k-1}(0, i^*)$ belongs. This word is well defined according to Assertion 3 of Theorem 1.

Theorem 2. The coding word $\mathbf{w} = \sigma_{1..N}(\mathbf{w}_N)$ is the reversal of the coding of the orbit of the face $(0, \mathbf{w}_N^*)$ under the action of \mathcal{E} . The vertices of the discrete segment with coding word \mathbf{w} are in a one-to-one correspondence with the faces of type \mathbf{w}_N of \mathcal{W} .

Proof. We write $\mathbf{w} = z_1 \cdots z_k \cdots z_{|\mathbf{w}|}$. We consider the orbit of $(0, \mathbf{w}_N^*)$ under the action of the exchange of pieces \mathcal{E} . The proof is done by induction on k . The property holds for $k = 1$: $(\mathbf{0}, \mathbf{w}_N^*)$ belongs to $E_1^*(\sigma_{1..N})(\mathbf{0}, z_{|\mathbf{w}|}^*)$. We assume that the induction hypothesis holds for all $\ell \leq k$ with $1 \leq k < |\mathbf{w}|$.

Hence $\mathcal{E}^{k-1}(\mathbf{0}, i^*) = ((M_{1..N})^{-1}\mathbf{l}(z_{|\mathbf{w}|-k+1} \cdots z_{|\mathbf{w}|}), \mathbf{w}_N^*)$ and $\mathcal{E}^{k-1}(\mathbf{0}, \mathbf{w}_N^*)$ is contained in $\mathcal{W}_{z_{|\mathbf{w}|-k}}$. Consequently, $\mathcal{E}^k(0, \mathbf{w}_N^*) = \mathcal{E}^{k-1}(0, \mathbf{w}_N^*) + M_{1..N}^{-1} \mathbf{e}_{z_{|\mathbf{w}|-k}}$ and $\mathcal{E}^{k-1}(\mathbf{0}, i^*) = (M_{1..N}^{-1}\mathbf{l}(z_1 \cdots z_{|\mathbf{w}|-k}), \mathbf{w}_N^*)$. The one-to-one mapping comes from $\langle -M_{1..N}^{-1}\mathbf{l}(z_1 \cdots z_k) + \mathbf{u}_N, \mathbf{v} \rangle = \langle -\mathbf{l}(z_1 \cdots z_k) + \mathbf{l}(\mathbf{w}), \mathbf{1} \rangle$.

Example 6. Let $\mathbf{u} = (2, 2, 3)$ on which Poincaré algorithm is applied, and let $\mathcal{C} = \text{cube}$ be the lower unit cube. One has $\mathbf{v} = (7, 13, 4)$.



The letters of $\mathbf{w} = 1231233$ correspond to the color of the faces of type 1 of \mathcal{W} ordered decreasingly by their scalar product with \mathbf{v} . The vertices of the discrete segment depicted in Example 3:

$$(0, 0, 0), (1, 0, 0), (1, 1, 0), (1, 1, 1), (2, 1, 1), (2, 2, 1), (2, 2, 2), (2, 2, 3)$$

are in one-to-one correspondence with the vertices of the faces of type 1 by the map

$$\mathbf{x} \mapsto -(M_{1..3})^{-1} \cdot \mathbf{x} + \mathbf{u}_N = \begin{pmatrix} -2 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 3 & -2 \end{pmatrix} \cdot \mathbf{x} + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}.$$

Remark 4. Theorem 2 does not only apply for \mathbf{w}_N but also for the other letters. Note that it allows a labelling of faces of a given type by increasing distance to the Euclidean plane with normal vector \mathbf{v} . Theorem 2 can be considered as an analogue of the description of Sturmian and Christoffel words in terms of codings of rotations acting on the unit circle. It also provides a second simple generation method for discrete segments.

5 Conclusion

We have described here a generation method for discrete segments connecting the origin to a given point $(u_1, u_2, u_3) \in \mathbb{N}^3$. We obtain two generation methods: the first one is stated in terms of an iteration of a finite number of substitutions governed by the choice of the underlying three-dimensional Euclid’s algorithm (see Section 3); the second one is of a more geometric flavor and involves a dual discrete plane (see Section 4.4). We recover here duality ideas that can be found in 7 in the framework of Christoffel words. Our contribution mostly relies in the

application and development of the formalism of [2] in the context of the study of discrete lines. Note that the use of generalized substitutions (see Section 2.2) associated with multi-dimensional continued fraction algorithms has also already proved its efficiency in discrete geometry for the generation of discrete planes, see [11,6]. We now aim at starting a thorough investigation and comparison of the generation properties of the most classical three-dimensional Euclid's algorithm.

References

1. Andres, E.: Discrete linear objects in dimension n : the standard model. *Graphical Models* 65, 92–111 (2003)
2. Arnoux, P., Ito, S.: Pisot substitutions and Rauzy fractals. *Bull. Belg. Math. Soc. Simon Stevin* 8(2), 181–207 (2001)
3. Arnoux, P., Mauduit, C., Shiokawa, I., Tamura, J.I.: Complexity of sequences defined by billiards in the cube. *Bull. Soc. Math. France* 122, 1–12 (1994)
4. Arnoux, P., Rauzy, G.: Représentation géométrique de suites de complexité $2n + 1$. *Bull. Soc. Math. France* 119, 199–215 (1991)
5. Berthé, V., Ferenczi, S., Zamboni, L.Q.: Interactions between dynamics, arithmetics and combinatorics: the good, the bad, and the ugly. In: *Algebraic and Topological Dynamics, Contemp. Math.*, vol. 385, pp. 333–364. Amer. Math. Soc., Providence (2005)
6. Berthé, V., Lacasse, A., Paquin, G., Provençal, X.: A study of Jacobi-Perron boundary words for the generation of discrete planes. Preprint (2010)
7. Berthé, V., de Luca, A., Reutenauer, C.: On an involution of Christoffel words and Sturmian morphisms. *European Journal of Combinatorics* 29, 535–553 (2008)
8. Brimkov, V.E., Barneva, R.P., Brimkov, B., de Vieilleville, F.: Offset approach to defining 3D digital lines. In: *Advances in Visual Computing*, pp. 678–687 (2008)
9. Brimkov, V.E., Barneva, R.P., Brimkov, B.: Minimal offsets that guarantee maximal or minimal connectivity of digital curves in nD . In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009. LNCS*, vol. 5810, pp. 337–349. Springer, Heidelberg (2009)
10. Broise-Alamichel, A., Guivarc'h, Y.: Exposants caractéristiques de l'algorithme de Jacobi-Perron et de la transformation associée. *Ann. Inst. Fourier (Grenoble)* 51(3), 565–686 (2001)
11. Fernique, T.: Generation and recognition of digital planes using multi-dimensional continued fractions. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) *DGCI 2008. LNCS*, vol. 4992, pp. 33–44. Springer, Heidelberg (2008)
12. Figueiredo, O., Reveillès, J.P.: New results about 3d digital lines. In: *Proc. Internat. Conference Vision Geometry V, Proc. SPIE*, vol. 2826, pp. 98–108 (1996)
13. Hubert, P.: Suites équilibrées. *Theoret. Comput. Sci.* 242, 91–108 (2000)
14. Klette, R., Rosenfeld, A.: Digital straightness—a review. *Discrete Applied Mathematics* 139, 197–230 (2004)
15. Lothaire, M.: *Algebraic Combinatorics on Words*. Cambridge University Press, Cambridge (2002)
16. Reveillès, J.P.: *Géométrie discrète, calcul en nombres entiers et algorithmique*. Thèse de Doctorat, Université Louis Pasteur, Strasbourg (1991)
17. Schweiger, F.: *Multidimensional Continued Fraction*. Oxford Univ. Press, New York (2000)
18. Toutant, J.L.: Characterization of the closest discrete approximation of a line in the 3-Dimensional space. In: *Advances in Visual Computing*, pp. 618–627 (2006)

Smooth 2D Coordinate Systems on Discrete Surfaces

Colin Cartade¹, Rémy Malgouyres¹, Christian Mercat², and Chafik Samir³

¹ Clermont-Université, LIMOS, Complexe des Cézeaux, 63172 Aubière, France
{colin.cartade,remy.malgouyres}@u-clermont1.fr

² Université Lyon 1, IUFM, Bât. Braconnier, 69622 Villeurbanne, France
christian.mercat@gmail.com

³ Clermont-Université, ISIT, Complexe des Cézeaux, 63172 Aubière, France
chafik.samir@u-clermont1.fr

Abstract. We introduce a new method to compute conformal parameterizations using a recent definition of discrete conformity, and establish a discrete version of the Riemann mapping theorem. Our algorithm can parameterize triangular, quadrangular and digital meshes. It can be adapted to preserve metric properties. To demonstrate the efficiency of our method, many examples are shown in the experiment section.

Keywords: parameterization, conformal, digital surfaces.

Introduction

Knowing a parameterization of a surface is very useful because it allows to work with functions instead of three dimensional sets. Thus, we can easily apply real analysis results to surfaces. Moreover, as parameterizations establish a correspondence between a surface and a part of the plane, we can then extend planar techniques to surfaces. For all these reasons, they are widely used in mesh processing: among the many applications, we can cite texture mapping, morphing, surface fitting, etc.

A parameterization should preserve the geometrical properties of the mesh: angles (conformal map), areas (authalic maps), lengths (isometric map), etc. But maps which are both conformal and authalic maps are isometric, and only developable surfaces have an isometric flat parameterization. In practice, people often look for conformal maps. They preserve angles, lengths ratios locally, and more generally the local aspect of the mesh. It is often sufficient to obtain a good parameterization.

In this paper we present a new algorithm to compute conformal parameterizations using the definition of discrete conformity given in [10]. Although, it was first used for meshes, with floating point coordinates, it has the main advantage of being easily adaptable to digital surfaces, with integer coordinates. We show that, in the case of triangular meshes, it is a generalization of the *cotan* conformal coordinates method [13] and establish a discrete version of the Riemann

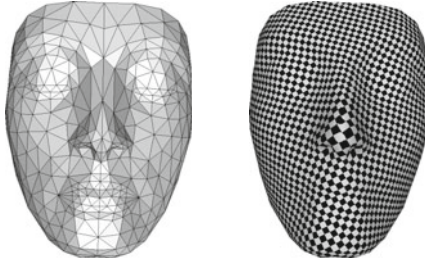


Fig. 1. Triangular mesh parameterization, $E = H + L'$ (see p. 67)

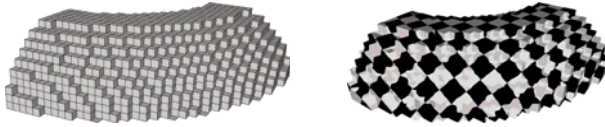


Fig. 2. Digital torus parameterization

mapping theorem. The boundary conditions introduced are closer to the real continuous theorem than those of classical conformal methods.

Our algorithm consists in minimizing a discrete energy to measure conformity. It can be generalized to preserve other properties such as faces areas and/or edges lengths allowing to obtain a more isometric parameterization. These energies can also be used to obtain parameterizations with free-boundary conditions as detailed in [4,8,9,14] and give better results around the boundary.

The rest of the paper is organized as follows. Section 1 introduces the definition of discrete conformal maps for quad meshes and show how it can be generalized to digital surfaces and triangular meshes. In Section 2 we explain how to choose boundary conditions, *i.e.* fix the position of some boundary points, to ensure uniqueness. We mainly focus on the two choices which lead to a discrete version of the Riemann mapping theorem and to the same parameterization as the *cotan* conformal coordinates. In Section 3, we describe precisely how we proceed in practice to compute parameterizations. Numerical illustrations are given in Section 4.

1 Discrete Conformal Parameterizations

1.1 Quad Meshes

In real continuous theory, a surface parameterization is a bijective application from the surface S in \mathbb{R}^3 to the plane: $(x, y, z) \in S \mapsto (s(x, y, z), t(x, y, z)) \in \mathbb{R}^2$. For meshes, it boils down to a point $v' = (s, t)$ assigned to each vertex $v = (x, y, z)$. In the sequel, we will identify v' with the complex number $s + it$.

Locally identifying each face (v_0, v_1, v_2, v_3) of a quad mesh to points in the plane (in one way or another) we can view the diagonals $v_2 - v_0$ and $v_3 - v_1$

as two complex numbers and compute the ratio $\rho = \frac{v_3 - v_1}{i(v_2 - v_0)}$, which is defined up to a global similarity. Following [10], we call this data a *discrete conformal structure* and we say that a parameterization is *discrete conformal* if it preserves the ratios ρ . In other words, for all faces of the mesh, we require that

$$\frac{v'_3 - v'_1}{v'_2 - v'_0} = i\rho. \quad (1)$$

Geometrically, such a parameterization preserves the angle between the diagonals and the ratio of their lengths. It is a property we expect since faces are small with respect to the whole mesh and a conformal map locally preserve angles and lengths ratios (its derivative is a similarity), For simplicity, we can rewrite (1) as a linear equation

$$v'_3 - v'_1 = i\rho(v'_2 - v'_0), \quad (2)$$

consequently a conformal parameterization can be seen as a solution of a (complex valued) linear system.

Remark 1. Even if the four vertices of a quad are not in the same plane we can define the ratio ρ . Indeed, the diagonals in \mathbb{R}^3 , when not colinear, can be viewed as two vectors spanning a plane, wherein the complex ratio can be computed. This choice amounts to defining the normal to the surface as the cross-product of these diagonals. A prior knowledge of the normal, therefore of the tangent plane, is another way to identify the quad-face to a quadrilateral in the complex plane, by projecting the vertices onto this tangent plane. The ratio does not depend on the choice of the normal basis identifying the tangent plane with the complex numbers. Together, all these identifications of the tangent plane at each quad, considered as local charts, form an *atlas* of the surface.

1.2 Triangular Meshes

In general, faces of meshes are not quads but triangles and the definition extends to this case: we add a new (combinatorial) *dual point* to each face and to each boundary edge, a standard procedure in remeshing. Then for each edge of the initial mesh we form a quad by joining the extremities of the edge and

- the two dual points inside the adjacent faces if it is not a boundary edge
- the dual points inside the adjacent face and on the edge if it is a boundary edge.

The construction is shown in Figure 3. On the left, we display the initial triangular mesh and the dual points, and on the right the new quad mesh.

By definition, quads consist of two triangles that do not necessarily belong to the same plane. To determine the ρ coefficient, we rotate one of them until it belongs to the plane of second, that is to say we flatten them using the *intrinsic* metric of the polyhedral surface. Once we have this quad structure and a ρ for each quad we can look for a parameterization in the same way as in Section 1.1. Thus we parameterize not only the initial vertices but also the dual points.

The use of the extrinsic or intrinsic distances does not seem to imply big differences as noted in another context in [2].

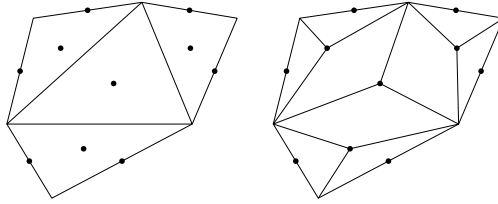


Fig. 3. Introduction of dual points

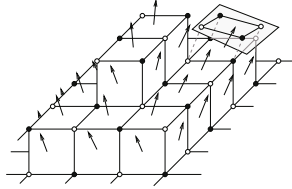


Fig. 4. Definition of ρ for digital surfaces

1.3 Digital Surfaces

The definition we gave in the previous section is not interesting as such when using digital surfaces whose faces are surfels. Indeed, these faces are planar squares and all the ρ coefficients equal 1. Therefore a more meaningful discrete conformal structure has to be defined, using *extrinsic* or non local data such as a given normal vector [11]: we compute a normal vector of each face using for instance the method described in [6], or coming from the scanned data. It allows us to determine the tangent plane of the surface in each surfel. Firstly, we project the four edges on this plane, obtaining a parallelogram which better approximates the continuous surface than the initial surfel. Secondly we define the ρ coefficient of a surfel as the one of this projected parallelogram. An example of the construction is depicted in Figure 4.

2 Boundary Conditions

2.1 Solutions of the System

The linear system (2) does not have a unique solution since there are more unknowns than equations: If we denote by n_f , n_e , n_b and n_v the number of faces, edges, border edges and vertices of the mesh, it consists of $2n_f$ real equations and has $2n_v$ real unknowns. We have on the one hand (Euler characteristic of the disc)

$$1 = n_f - n_e + n_v, \quad (3)$$

and on the other hand (mesh property)

$$4n_f = 2n_e - n_b. \quad (4)$$

Adding $2 \times$ (3) to (4) we obtain

$$2(n_v - n_f) = n_b + 2.$$

Hence, in order to ensure uniqueness we need $n_b + 2$ real constraints.

2.2 Connection to Real Continuous Theory

The Riemann mapping theorem states that each surface homeomorphic to the closed disc is in fact conformally equivalent to the closed disc. Besides the holomorphic map is unique if one boundary is mapped to the other one and the images of 3 boundary points are fixed [116,317].

In the same way, we can map the boundary of the mesh to the unit circle. Thus we obtain n_b additional real constraints. Then, if we fix the images of two of the boundary vertices, we have the

$$(n_b - 2) + (2 \times 2) = n_b + 2$$

real constraints we are looking for.

This leads to the following discrete version of Riemann mapping theorem: if two (almost three) boundary vertices are fixed, there exists only one discrete conformal parameterization whose boundary points belong to the unit circle. These boundary points are not different from other boundary points. Our boundary conditions are much closer to the Riemann theorem than those of other classical discrete conformal algorithms: [513] fix all the boundary points and [49] fix two boundary points (that accumulate conformal distortion) but the other ones are not mapped on the circle.

2.3 Connection to the *cotan* Conformal Coordinates

In this section, we will show that our method is a generalization of parameterizations with the *cotan* conformal coordinates [413]. We remind the reader that this method applies to triangular meshes and consists in

1. fixing the images of the boundary points, often on a convex boundary,
2. solving the following system:
for each vertex v_i which is not on the boundary of the mesh

$$\sum_{j : v_j \text{ neighbour of } v_i} (\cot \alpha_{ij} + \cot \alpha_{ji})(v'_j - v'_i) = 0 \quad (5)$$

(the angle α_{ij} and α_{ji} are defined on Figure 5, left picture).

Suppose that in the construction of dual points described in Section 1.2 we add the circumcenter of the triangles and the middle of the boundary edges. Consider two adjacent triangles faces (v_0, v_1, v_2) and (v_0, v_3, v_1) . We denote by v_3 and v_4 their circumcenters and by c the middle of $[v_0, v_1]$. An example of the

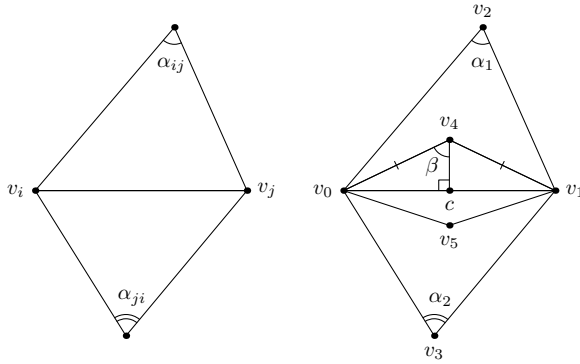


Fig. 5. Definition of the angles in the *cotan* formula

construction is shown in Figure 5 (right picture). We want to compute the ρ coefficient of the quad (v_0, v_5, v_1, v_4) .

First, since the angle in c is right, ρ is real:

$$\rho = \frac{\|v_4 - c\|}{\|v_1 - v_0\|} + \frac{\|v_5 - c\|}{\|v_1 - v_0\|}.$$

Second,

$$\frac{\|v_4 - c\|}{\|v_1 - v_0\|} = \cot \beta = \cot \alpha_1.$$

since the triangle (v_0, c, v_4) is right in c (first equality) and $\alpha_1 = \frac{v_0 v_4 v_1}{2}$ according of the inscribed angle theorem (second equality).

Finally

$$\rho = \cot \alpha_1 + \cot \alpha_2.$$

Note that the coefficients are the same as in (5).

We denote by $\rho(v_i, v_j)$ the ρ coefficient of the quad containing the diagonal $[v_i, v_j]$. Adding the equations in (2) involving a particular initial vertex v_i we obtain

$$\sum_{j : v_j \text{ neighbour of } v_i} \rho(v_i, v_j)(v'_j - v'_i) = 0$$

It is in fact the sum over the dual edges which form a loop. Hence the system (2) is equivalent to a system which has (5) as a subsystem.

Due, to the results of section 2.1 we have to add $\frac{n_b}{2} + 1$ real constraints to ensure uniqueness. We choose to fix the initial boundary points (those of the triangular mesh) and one of the dual boundary points. Hence the coordinates v'_i of the initial mesh satisfy the same linear system as with *cotan* conformal coordinates and we obtain the same solution.

We have proved that our method is a generalisation with arbitrary dual points and boundary constraints of the *cotan* conformal coordinates. It can be of interest when some angles of the triangles are obtuse. Then the circumcenters are

not necessarily inside the triangles and the coefficients in (5) can be negative. Thus conditions of Tutte theorem [15] are not verified and the *cotan* conformal coordinates method can fail.

3 Practical Computation

3.1 Energy Minimization

Many parameterizations methods, including [4,5,9,13], consist in solving sparse linear systems. As the system of equations (2) is also sparse, we could think of using similar techniques. But the boundary condition, *i.e.* remaining on a circle, is not linear and even not quadratic. That is why we implement a non-linear minimization technique.

We denote by $\rho(v_i, v_j)$ the ρ coefficient of the face containing the diagonal $[v_i, v_j]$. Then we introduce the conformal energy

$$H = \sum |(v'_i - v'_j) - \rho(v_i, v_k)(v'_k - v'_i)|^2$$

where the sum is over all the quads (v_i, v_j, v_k, v_l) of the mesh, and the boundary energy

$$C = \sum (|v'_i|^2 - 1)^2$$

where the sum is over all the boundary vertices v_i except the two ones whose parameters are fixed. We search the parameters v'_i which minimize the total energy

$$E = \alpha H + \beta C$$

for chosen positive real numbers α and β . The minimization is performed using a conjugate gradient method (Fletcher-Reeves algorithm, [12]).

3.2 Initial Conditions

The minimization algorithms we use are guaranteed to converge to a local minimum but not necessarily to the global one. It is a very important issue in this case since there is no uniqueness in the Riemann theorem if the map is not one to one. For example, the one to one parameterization of the unit disc with three fixed points is the identity map. However, with convenient initial conditions our algorithm can perfectly lead to a discrete approximation of the $z \mapsto z^2$ map.

In practice, it works quite well to set at the beginning the images of the boundary points on a circle with the same distances between them as on the boundary of the mesh. And we set the initial positions of the images of the interior points at the origin. The algorithm more or less acts like the relaxation of a network of springs. We must also mention that in order to have less distortion it is generally better to choose the fixed points as far as possible from each other. For example when we fix two points, we try to select points with around half of the boundary length between them. An example is given in Figure 6 where the fixed points are represented by big dots.

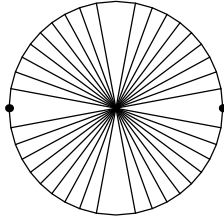


Fig. 6. Example of initial configuration

3.3 Preservation of Lengths and Areas

Yet, we focused on computing conformal maps to preserve angles and thus shapes locally. It is certainly a key feature in mesh parameterization but not enough to ensure a good result. Indeed, conformal maps can lead to parameterizations which are very tight in some regions and more sparse in others. If we map a checkerboard with such a parameterization we obtain big squares in the first regions and little ones in the others which is of course unsatisfactory.

To avoid these artifacts we define new energies attached to preserving metric properties such as:

1. the area of the faces,
2. the length of the edges.

First, we introduce the *authalic* energy

$$A = \sum \left(\operatorname{Im} \left((v'_k - v'_i) \overline{(v'_l - v'_j)} \right) \right)^2 - \left(\|(v_l - v_i) \wedge (v_k - v_i)\| + \|(v_k - v_i) \wedge (v_j - v_i)\| \right)^2$$

where the sum is over all the quads (v_i, v_j, v_k, v_l) of the mesh, and we secondly define the *metric* energy

$$L = \sum \left(|v'_i - v'_j|^2 - \|v_i - v_j\|^2 \right)^2 \quad (6)$$

where the sum is over all the edges $[v_i, v_j]$. Then, we consider the energy

$$E = \alpha H + \beta A + \gamma L.$$

There are many more equations and we are no longer looking for a unique solution of a system but rather for the minimum of an energy. However, as any isometric transformation of a given parameterization has the same energy, we ensure uniqueness by fixing the image of one boundary point and the direction of the next boundary edge.

Note that such a minimal energy parameterization will not be conformal, unless the surface is developable. Indeed a map preserving both angles and areas is isometric and only developable surface have an isometric flat parameterization.

But choosing the coefficient α , β and γ conveniently we can obtain more accurate results. Moreover, as there is no longer a condition on the boundary we also obtain a more “natural” boundary, adapted to the mesh. The choice moved from the boundary points to the coefficients.

It can be quite slow to compute the energy L (and also A). To speed up our algorithm, we can minimize only the distortion of the metric along the boundary. Thus we introduce the metric energy L' defined as (6) where the sum is over the boundary edges only. It allows us to obtain a conformal map with a more “natural” boundary, closer to another classical Riemann-Hilbert condition (16). Note also that even with the initial conditions described in the previous section, in general, for numerical reasons, the algorithm does not converge towards the right local minimum if we do not use the energy C . It does, however, if we use the following two steps process:

1. minimize H with a fixed boundary.
2. use this minimum as initial condition to minimize E .

Moreover, as the minimization is very fast when we fix all the boundary points, it also speeds up the convergence.

4 Results

4.1 Comparison of the Energies

We first computed parameterizations of the mesh $(s, t) \mapsto \cos(s) + \cos(t)$ (discretized with quad faces) to show the main advantages and drawbacks of each of the energies C (circle), L' (length of the boundary), L (length) and A (area). You can see the corresponding results in Figure 7, 8, 9 and 10. Additionally, the algorithm can be applied on triangular meshes as shown in Figure 11.

The parameterizations are displayed on the left. On the right, we mapped a 32×32 checkerboard on the surface using this parameterization. With the circle boundary energy $E = H + C$ (Figure 7), we obtain a conformal parameterization (the shape of the squares is preserved) but the behavior around the boundary is not natural (because a circle is very different from the true boundary of the mesh). In particular there is a big distortion of the metric in this region. If we use the boundary metric energy L' instead of C (Figure 8), the map is still conformal but the boundary is better preserved. We observed that the results are visually quite close to those of other methods with free boundary (ABF, circle patterns, etc.). The use of the energies L and A has the same effect on the boundary. Besides, even if the parameterization is not conformal (see all the red points in Figure 9), the texture mapping looks more accurate since all the squares have the same size. However, close inspection shows that some squares of the checkerboard become rectangles after the mapping.

4.2 Digital Surfaces

We also computed parameterizations of digital surfaces. We used a digital torus consisting of around 1500 surfels. We first computed a parameterization using

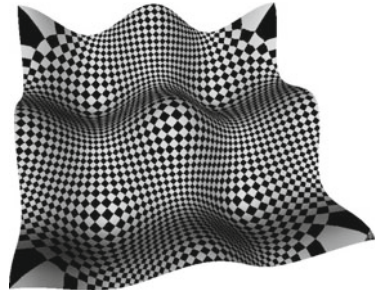
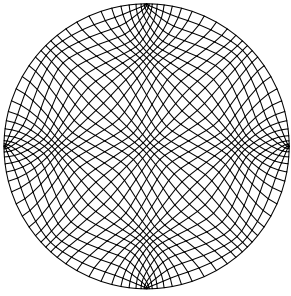


Fig. 7. $E = H + C$, circle boundary

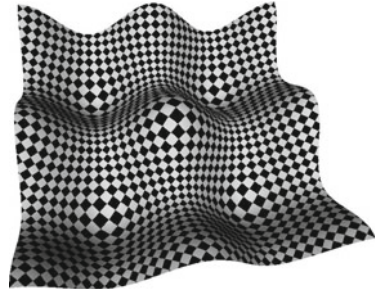
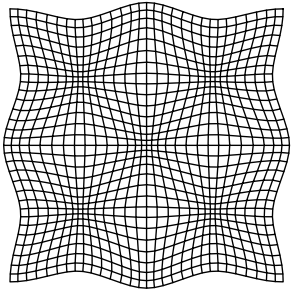


Fig. 8. $E = H + L'$, more isometric along the boundary

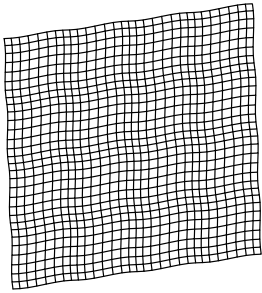


Fig. 9. $E = H + L$, more isometric

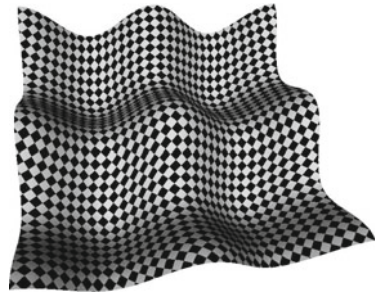
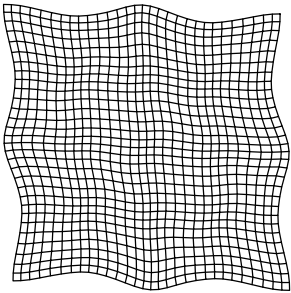


Fig. 10. $E = H + A$, more area preserving

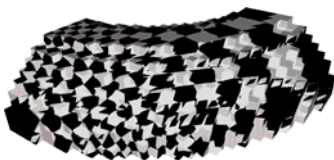


Fig. 11. Method for quad meshes with digital normal, $E = H + C$



Fig. 12. Digital method with smoothed normal, (a) $E = H + C$, (b) $E = H + A$

the method for quad meshes without smoothing the normal. As expected the resulting texture mapping is not good: we do not even distinguish the checkerboard squares in some regions of Figure 11. Then we use our method for digital surfaces and obtain better results: we clearly distinguish the checkerboard on Figure 12. This figure also shows that the influence of additional energies is the same as with float meshes.

5 Conclusion and Future Work

We have described a new method of conformal parameterization that can be applied to different meshes, including triangular meshes and digital surfaces. As a proof of concept, we have used different cost functions whose minimum preserves more or less the shape, the size, and the boundary conditions. An important feature of our approach is the introduction of a recent definition of discrete conformity allowing many possibilities for more accurate results. We have shown various experimental results to illustrate the different possibilities. In future work, we will study the choice of coefficients α , β , etc. In particular, it would be interesting to compute the more isometric conformal parameterization.

Acknowledgement. This work was partially supported by the ANR project KIDICO (ANR-2010-BLAN-0205-02).

References

1. Ahlfors, L.V.: Complex analysis, 3rd edn. McGraw-Hill Book Co., New York (1978); an introduction to the theory of analytic functions of one complex variable, International Series in Pure and Applied Mathematics
2. Bobenko, A.I., Mercat, C., Schmieß, M.: Conformal Structures and Period Matrices of Polyhedral Surfaces. In: Computational Approach to Riemann Surfaces. Springer, Heidelberg (2011)

3. Bobenko, A.I., Schröder, P., Sullivan, J.M., Ziegler, G.M. (eds.): Discrete differential geometry, Oberwolfach Seminars, vol. 38. Birkhäuser, Basel (2008), <http://dx.doi.org/10.1007/978-3-7643-8621-4> (papers from the seminar held in Oberwolfach, May 30–June 5, 2004)
4. Desbrun, M., Meyer, M., Alliez, P.: Intrinsic parameterizations of surface meshes. *Computer Graphics Forum* 21, 209–218 (2002)
5. Floater, M.S.: Mean value coordinates. *Computer Aided Geometric Design* 20(1), 19–27 (2003)
6. Fourey, S., Malgouyres, R.: Normals estimation for digital surfaces based on convolutions. *Computers & Graphics* 33(1), 2–10 (2009)
7. Gu, X.D., Yau, S.T.: Computational conformal geometry, *Advanced Lectures in Mathematics (ALM)*, vol. 3. International Press, Somerville (2008); with 1 CD-ROM (Windows, Macintosh and Linux)
8. Kharevych, L., Springborn, B., Schröder, P.: Discrete conformal mappings via circle patterns. *ACM Transactions on Graphics (TOG)* 25(2), 438 (2006)
9. Lévy, B., Petitjean, S., Ray, N., Maillot, J.: Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics* 21(3), 362–371 (2002)
10. Mercat, C.: Discrete Riemann surfaces and the Ising model. *Communications in Mathematical Physics* 218(1), 177–216 (2001)
11. Mercat, C.: Discrete complex structure on surfel surfaces. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) *DGCI 2008. LNCS*, vol. 4992, pp. 153–164. Springer, Heidelberg (2008)
12. Nocedal, J., Wright, S.J.: *Numerical optimization*. Springer, Heidelberg (1999)
13. Pinkall, U., Polthier, K.: Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics* 2(1), 15–36 (1993)
14. Sheffer, A., de Sturler, E.: Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers* 17(3), 326–337 (2001)
15. Tutte, W.: Convex representations of graphs. *Proceedings of the London Mathematical Society* 3(1), 304 (1960)
16. Wegert, E.: Nonlinear Riemann-Hilbert problems—history and perspectives. In: *Computational Methods and Function Theory 1997 (Nicosia)*, Ser. Approx. Decompos., vol. 11, pp. 583–615. World Sci. Publ., River Edge (1999)

An Improved Riemannian Metric Approximation for Graph Cuts

Ondřej Daněk and Pavel Matula

Centre for Biomedical Image Analysis, Faculty of Informatics
Masaryk University, Brno, Czech Republic
{xdanek2,pam}@fi.muni.cz

Abstract. Boykov and Kolmogorov showed that it is possible to find globally minimal contours and surfaces via graph cuts by embedding an appropriate metric approximation into the graph edge weights and derived the requisite formulas for Euclidean and Riemannian metrics [3]. In [9] we have proposed an improved Euclidean metric approximation that is invariant under (horizontal and vertical) mirroring, applicable to grids with anisotropic resolution and with a smaller approximation error. In this paper, we extend our method to general Riemannian metrics that are essential for graph cut based image segmentation or stereo matching. It is achieved by the introduction of a transformation reducing the Riemannian case to the Euclidean one and adjusting the formulas from [9] to be able to cope with non-orthogonal grids. We demonstrate that the proposed method yields smaller approximation errors than the previous approaches both in theory and practice.

1 Introduction

Combinatorial optimization using graph cuts presents a powerful energy minimization tool that has become very popular in the recent years in computer vision and image processing fields, mainly for its efficiency and applicability to a wide range of problems [5]. For instance, modern approaches to image segmentation often express the problem in terms of minimization of a suitable energy functional and can be solved via graph cut based algorithms if the energy is graph representable [11,2]. The graph cut segmentation framework has several advantages over traditional methods such as applicability to N-D problems, straightforward integration of various types of regional or geometric constraints or the ability to find a global minimum in polynomial time [4,2].

Many of the energy functionals used in computer vision such as the Chan-Vese segmentation model [8] or the geodesic active contours [7] require minimization of a length dependent term, e.g. length of the segmentation boundary under a given metric (Euclidean in the former example and Riemannian in the latter one). In order to minimize such energies using graph cuts the metric approximation has to be embedded into the graph. A seminal paper in this area is due to Boykov and Kolmogorov [3] who showed that graph cuts can be viewed as hypersurfaces and that it is possible to compute geodesics and globally minimal surfaces via

graph cuts for Euclidean and Riemannian metrics with arbitrarily small error depending primarily on the size of the neighbourhood system that is used to connect the nodes in the graph.

In [9] we have devised an improved Euclidean metric approximation that is invariant under image mirroring, applicable to images with anisotropic resolution and with a smaller approximation error, especially for small neighbourhoods that are used in computer vision (large neighbourhoods result in better approximation but also longer computation and high memory consumption). We have presented the practical benefits of our method on the graph cut based minimization of the Chan-Vese segmentation model [16]. In this paper, we extend our method to general Riemannian metrics, making it applicable to the graph cut based geodesic segmentation model [3] or stereo matching [14]. To achieve this, we set up a transformation matrix that projects a Riemannian space with a locally constant metric tensor onto the Euclidean plane. Subsequently, due to the linearity of the transformation, we are able to exploit the Cauchy-Crofton based formulas from [9] to derive the edge weights approximating the Euclidean metric in the transformed space and obtain a good approximation for the original problem. Nevertheless, the formulas are adjusted to be able to cope with non-orthogonal grids that may arise during the transformation. To demonstrate that this method gives a better approximation than [3] we plot and compare the approximation error it yields for lines under different angular orientations and also show practical examples on the graph cut based geodesic segmentation.

To conclude the introduction, we also refer the reader to recent methods based on continuous maximum flows [13] allowing to find globally optimal surfaces for both isotropic and anisotropic Riemannian metrics [11,15] minimizing the metrication artefacts, i.e. offering superior results compared to the traditional combinatorial graph cuts (dealt with in this paper) in many situations.

This paper is organized as follows. In Section 2 we state the problem and briefly describe the concept of cut metrics including current approaches to Euclidean metric approximation via graph cuts. In Section 3 we present the state-of-the-art approach to Riemannian metric approximation and give details about the proposed method extending our previous work. The comparison of both methods is available in Section 4 and in Section 5 we provide discussion on their complexity and computational demands. We conclude the paper in Section 6.

2 Preliminaries

In this section we review the concept of cut metrics and explain the current approaches to Euclidean metric approximation via graph cuts. For simplicity we consider only the 2D case in the following text. However, the methods are directly extensible to 3D and this extension is covered in Section 3.4.

2.1 Cut Metrics

Given a connected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ a cut \mathcal{C} is a partitioning of the graph nodes \mathcal{V} into two disjoint subsets S and T such that $\mathcal{V} = S \cup T$. It is defined

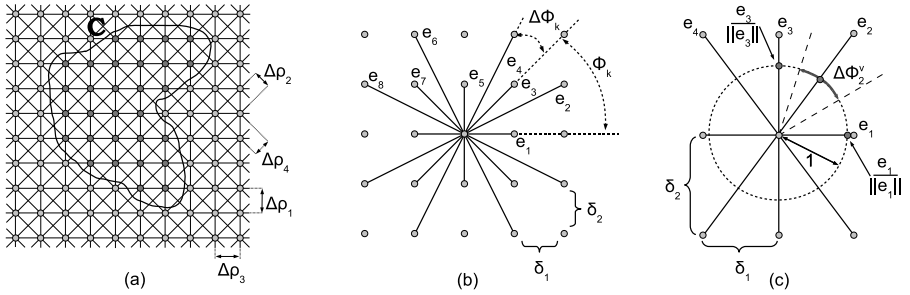


Fig. 1. (a) A grid graph with 8-neighbourhood system, $\Delta\rho_k$ denotes the distance between the closest lines generated by vector e_k . (b) 16-neighbourhood system and the computation of $\Delta\phi_k$ (c) 8-neighbourhood system and the computation of $\Delta\phi_k^v$.

by the set of edges connecting the nodes in S with nodes in T . The cost of a cut $|\mathcal{C}|_{\mathcal{G}}$ is the sum of the weights of its edges. Lets assume \mathcal{G} is embedded in a regular orthogonal grid. Example of such graph is depicted in Fig. 1. In [3] Boykov and Kolmogorov showed that cuts in grid graphs can be viewed as hypersurfaces (in our case as contours) and vice versa. Looking at Fig. 1a we can see that the closed contour C naturally defines a cut in \mathcal{G} by partitioning the graph nodes into two groups S and T depending on whether they lie inside or outside the contour. Alternatively, the cut induced by the contour can be defined as the set of edges it severs¹. Due to this geometric meaning of cuts we can naturally interpret the cost of a cut as the “length” of a corresponding contour.

Now, imagine the image segmentation problem. In graph cut based image segmentation [2] the input image is converted to a graph where every node corresponds to an image voxel and the nodes are connected according to a chosen neighbourhood system. A grid graph is obtained where every cut can be viewed as a binary (or foreground/background) segmentation of the image as they both partition the image into two parts. Moreover, if the edge weights are chosen appropriately, the cost of every cut can resemble the energy of a chosen energy functional (e.g. the Chan-Vese model [8,16]) of the corresponding segmentation. Thus, minimizer of the energy can be found by finding a *minimal cut* in the graph. This is a fast operation that can be performed in polynomial time [4].

However, as mentioned earlier many of the energy functionals [7,8] contain a length dependent term. Usually, the length of the segmentation boundary is being minimized under a chosen metric. Hence, a metric approximation has to be embedded into the graph in order to optimize such energy functional via graph cuts. Because we have already explained that the cost of a cut can be interpreted as the length of a corresponding contour (in this case, the segmentation boundary plays the role of the contour) all we have to do is to choose the edge weights appropriately such that for any cut its cost approximates the length of the corresponding contour under a chosen metric. In [3] Boykov and

¹ Obviously, for a grid with a finite resolution the correspondence between a cut and a contour is not unique and multiple contours can be found representing the same cut.

Kolmogorov showed that this can be achieved with arbitrarily small error and devised the requisite edge weight formulas for Euclidean and Riemannian metrics. The complete set of metrics that can be approximated via graph cuts has then been described in [10].

2.2 Euclidean Metric Approximation

Lets assume the graph \mathcal{G} is embedded in a regular orthogonal 2D grid with all nodes having topologically identical neighbourhood system and with δ_1 and δ_2 determining the spacing between the nodes in horizontal and vertical direction, respectively. An example of such a graph with 8-neighbourhood system and $\delta_1 = \delta_2$ is depicted in Fig. 1a. Further, let the neighbourhood system \mathcal{N} be described by a set of vectors $\mathcal{N} = \{e_1, \dots, e_m\}$. We assume that the vectors are listed in the increasing order of their angular orientation $0 \leq \phi_k < \pi$. We also assume that vectors e_k are undirected (we do not differentiate between e_k and $-e_k$) and shortest possible in given direction, e.g. 16-neighbourhood is represented by a set of 8 vectors $\mathcal{N}_{16} = \{e_1, \dots, e_8\}$ as depicted in Fig. 1b. Finally, we define the distance between the nearest lines generated by vector e_k in the grid as $\Delta\rho_k$ (for the 8-neighbourhood system these are depicted in Fig. 1a).

Assuming to each edge e_k is assigned a particular weight $w_k^\mathcal{E}$ (this weight is the same for all nodes), imagine we are given a regular curve C as shown in Fig. 1a. Further, lets assume the cut induced by the contour is defined by the set of edges it intersects. The question is how to set the edge weights so that the capacity of the cut $|C|_\mathcal{G}$ approximates the Euclidean length $|C|_\mathcal{E}$ of the contour. The method of [3,9] is based on the Cauchy-Crofton formula that links Euclidean length $|C|_\mathcal{E}$ of a contour C with a measure of a set of lines intersecting it:

$$|C|_\mathcal{E} = \frac{1}{2} \int_{\mathcal{L}} n_c(l) dl, \quad (1)$$

where \mathcal{L} is the space of all lines and $n_c(l)$ is the number of intersections of line l with the contour C . Because every line in a plane is uniquely identified by its angular orientation ϕ and distance ρ from the origin the formula can be rewritten in the form:

$$|C|_\mathcal{E} = \int_0^\pi \int_{-\infty}^{+\infty} \frac{n_c(\phi, \rho)}{2} d\rho d\phi, \quad (2)$$

discretizing the formula and following the derivation steps of [3,9] we end up with the following edge weights:

$$w_k^\mathcal{E} = \frac{\Delta\rho_k \Delta\phi_k}{2}, \quad (3)$$

where, as pointed out in [3], the distance between the closest lines generated by vector e_k in the grid can be calculated by dividing the area of the grid cell by the length of e_k :

$$\Delta\rho_k = \frac{\delta_1 \delta_2}{\|e_k\|}. \quad (4)$$

According to [3] using these weights $|\mathcal{C}|_{\mathcal{G}}$ converges to $|\mathcal{C}|_{\mathcal{E}}$ as $\delta_1, \delta_2, \sup_k \Delta\phi_k$ and $\sup_k \|e_k\|$ get to zero. Namely the value of $\sup_k \Delta\phi_k$ is important as it can be easily controlled by the choice of the neighbourhood. For dense neighbourhoods a very good approximation may be obtained [9].

Unfortunately, due to the computational demands small neighbourhoods are used in practice. For this reason, we have introduced an improved method in [9] having a smaller approximation error. It consists in replacing $\Delta\phi_k$ in Eq. 3 with:

$$\Delta\phi_k^v = \frac{\Delta\phi_k + \Delta\phi_{k-1}}{2}. \quad (5)$$

This modification attempts to calculate a better partitioning of the space of all angular orientations among the vectors e_k . In 2D this space is represented by a unit circle and points $\frac{e_k}{\|e_k\|}$ lying on the circle can be treated as samples from this space. To partition the space among these points a Voronoi diagram is computed on the unit circle and $\Delta\phi_k^v$ is chosen as the length of the Voronoi cell (circular arc) corresponding to $\frac{e_k}{\|e_k\|}$. Alternatively, $\Delta\phi_k^v$ is the measure of the lines closest to e_k in terms of their angular orientation. The whole construction is depicted in Fig. 1c and reduces to the Eq. 5 in 2D. Moreover, besides having a smaller approximation error this construction is also invariant under (horizontal and vertical) mirroring and directly generalizable to higher dimensions which the original approach of [3] from Fig. 1b is not.

3 Riemannian Metrics

3.1 Introduction

Giving all the theory on Riemannian geometry is beyond the scope of this paper, therefore we confine ourselves only to the basic notation required for proper understanding of the rest of this section. In Riemannian geometry each point of the space is associated with a *metric tensor* M that controls how an inner product of two vectors is calculated. This tensor is a symmetric positive definite matrix (a bilinear form) that varies smoothly over the space. In case M is constant the Riemannian norm of a vector u is calculated as:

$$\|u\|_{\mathcal{R}} = \sqrt{u^T \cdot M \cdot u} \quad (6)$$

A standard Euclidean norm is obtained when M equals to the identity matrix. In the non-constant case the distance between two points in a Riemannian space does depend on their displacement from the origin, as opposed to the Euclidean geometry. However, full understanding of this case is not necessary for the oncoming derivations.

Geometrical interpretation of the metric tensor is intuitive and expresses local contraction or dilation of the space. The properties follow from the spectral decomposition of M . Because M is symmetric positive definite it has two real-valued eigenvalues λ_1 and λ_2 and two orthogonal eigenvectors. The space is then “stretched” in the directions corresponding to the eigenvectors by the value of

$\sqrt{\lambda_1}$ and $\sqrt{\lambda_2}$. This behaviour has interesting utilizations in the image processing field. In [7] Caselles et al. showed that the image segmentation problem can be solved by finding the shortest path (geodesic) in a Riemannian space with an image derived metric tensor. In each voxel the tensor dilates the space in the direction of image gradient making the segmentation follow edges and object boundaries. Another interesting application is in stereo matching [14].

3.2 State of the Art

Until the seminal work of Boykov and Kolmogorov [3] the geodesic segmentation problem was solved solely using the level set framework [12]. In [3] Boykov and Kolmogorov showed that in some sense graph cuts can be treated as a combinatorial counterpart of the level set method, both having implicit boundary representation (see also [2]). Using integral geometry they derived the edge weight formulas for Euclidean as well as general Riemannian metrics allowing graph cut based solution to the geodesic segmentation problem. The formulas have the following form in 2D:

$$w_k^{\mathcal{R}} = w_k^{\mathcal{E}} \cdot \frac{\det M}{(u_k^T \cdot M \cdot u_k)^{3/2}}, \quad (7)$$

where u_k is a unit vector in the direction of e_k . As can be seen, in their approach the edge weights for a Riemannian metric are obtained by multiplying the Euclidean metric weights by a coefficient depending on the metric tensor M and the direction of e_k (the second term vanishes when M is the identity matrix). In the following subsection we propose a different method and show that it has a smaller approximation error.

3.3 Proposed Method

Like in the Euclidean case, let us assume the graph \mathcal{G} is embedded in a regular orthogonal 2D grid with all nodes having topologically identical neighbourhood system $\mathcal{N} = \{e_1, \dots, e_m\}$. Examples of \mathcal{N}_8 and \mathcal{N}_4 grid graphs are depicted in Fig. 1a and Fig. 2a, respectively. In addition, we assume that the graph is placed in a Riemannian space with the metric tensor sampled in the graph nodes, i.e. each graph node p is associated with a metric tensor $M(p)$ (e.g. the image derived metric tensor [3]). Without loss of generality we also assume that the spacing between the graph nodes is 1 in all directions. Because $M(p)$ already defines space stretching it is possible to embed the grid resolution right into the matrix M (more on this topic later in this section). Finally, our aim is to set the edge weights $w_k^{\mathcal{R}}$ such that for any contour C the cost of the corresponding cut $|C|_{\mathcal{G}}$ (i.e. the sum of the weights of the edges the contour severs) approximates the Riemannian length of the contour $|C|_{\mathcal{R}}$.

Choosing one of the graph nodes, let us assume the metric tensor $M(p)$ (we will omit the node specification in the following text) is locally constant around p . Further, recall that the geometric interpretation of M is space dilation and that the space is dilated by the amount corresponding to the square root of the

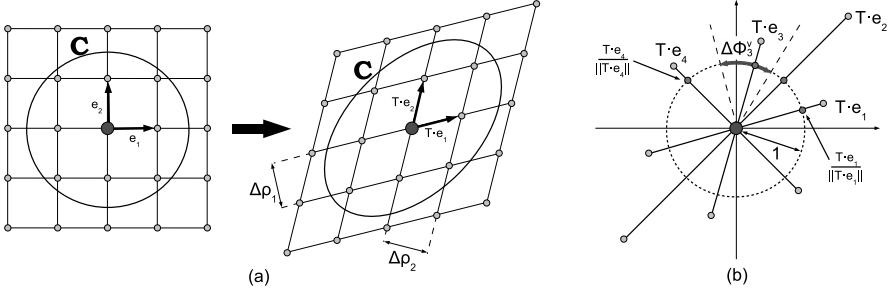


Fig. 2. (a) A grid graph with 4-neighbourhood system and a circular contour and the effect of space transformation by matrix T . $\Delta\rho_k$ is the distance between the closest lines generated in the transformed grid by the transformed vectors e_k . (b) 8-neighbourhood system transformed by the matrix T and the computation of $\Delta\phi_k^v$.

eigenvalues of M . To reflect this, we set up a symmetric transformation matrix T having the same eigenvectors as M but with eigenvalues being the square root of those of M . We have that:

$$M = T^T \cdot T. \quad (8)$$

Subsequently, the matrix T can be used to project the Riemannian space with a locally constant metric tensor onto the Euclidean plane. It holds that measuring Euclidean distances in the transformed space is equivalent to measuring Riemannian distances in the original space.

Lemma 1. *Given two points u and v , a constant metric tensor M and the corresponding transformation matrix T , it holds that the Euclidean distance of $T \cdot u$ and $T \cdot v$ equals to the Riemannian distance of u and v .*

$$\begin{aligned} \text{Proof. } d_{\mathcal{E}}(T \cdot u, T \cdot v) &= \|T \cdot (u - v)\|_{\mathcal{E}} = \sqrt{(T \cdot (u - v))^T \cdot (T \cdot (u - v))} = \\ &= \sqrt{(u - v)^T \cdot T^T \cdot T \cdot (u - v)} = \sqrt{(u - v)^T \cdot M \cdot (u - v)} = \|u - v\|_{\mathcal{R}} = \\ &= d_{\mathcal{R}}(u, v) \quad \square \end{aligned}$$

The effect of the space transformation using the matrix T on the local neighbourhood of graph node p is illustrated in Fig. 2a. Depending on T we may obtain a skewed non-orthogonal grid. However, note that due to the linearity of the transformation, the number of intersections between the grid and the contour C is preserved. This has an important corollary – if the edge weights are chosen so that the cut cost approximates the Euclidean length in the transformed space we would obtain also approximation of the original Riemannian length.

The last question remains, how to set up the edge weights so that the cut cost approximates the Euclidean length in the transformed space. Because the space is Euclidean, Eq. 3 based on the Cauchy-Crofton formula for Euclidean spaces is sufficient to obtain the edge weights. The only difference is the core set of lines used to compute $\Delta\rho_k$ and $\Delta\phi_k^v$ where the lines generated by the transformed neighbourhood system have to be considered. Because the transformed grid is

potentially non-orthogonal, Eq. 4 can no longer be used. However, the distance between the closest lines generated in the grid by vector $T \cdot e_k$ can still be calculated by dividing the area of the grid cell by the length of $T \cdot e_k$:

$$\Delta\rho_k = \frac{\det T}{\|T \cdot e_k\|_{\mathcal{E}}} = \frac{\sqrt{\det M}}{\|e_k\|_{\mathcal{R}}}. \quad (9)$$

The calculation of $\Delta\phi_k^v$ remains the same, i.e. using a Voronoi diagram on a unit hypersphere (circle in 2D) the measure of angular orientations closest to $T \cdot e_k$ is computed as illustrated in Fig. 2b. Alternatively, in 2D it can be obtained by averaging the angles between $T \cdot e_k$ and its two neighbours.

So far, we have considered only local behaviour of the Riemannian space. To obtain a generalization for the general case with varying metric tensor a different metric tensors $M(p)$ is used to compute the edge weights according to the derived formulas for each graph node p . Thus, w_k is no longer the same across all nodes as in the Euclidean case. Finally, notice that the derived formulas are in accordance with our previous results. In particular, the Euclidean space with node spacing δ_1 and δ_2 from Section 2.2 can be simulated using the following constant metric tensor and the corresponding transformation matrix:

$$M = \begin{pmatrix} \delta_1^2 & 0 \\ 0 & \delta_2^2 \end{pmatrix} \quad T = \begin{pmatrix} \delta_1 & 0 \\ 0 & \delta_2 \end{pmatrix}, \quad (10)$$

where M embeds the space stretching along x and y axis of the coordinate system by δ_1 and δ_2 , respectively. Using this matrix Eq. 9 reduces to Eq. 4 and the same edge weights are derived.

3.4 Extension to 3D

The Cauchy-Crofton based edge weights have the following form in 3D [39]:

$$w_k^{\mathcal{E}} = \frac{\Delta\rho_k \Delta\phi_k^v}{\pi}. \quad (11)$$

To compute $\Delta\rho_k$ Eq. 9 may be used as it is. The space of all angular orientations is represented by a unit sphere surface in 3D. The sphere surface area has to be partitioned among the points $\frac{T \cdot e_k}{\|T \cdot e_k\|}$ using a spherical Voronoi diagram [6] to obtain $\Delta\phi_k^v$.

4 Experimental Results

In this section, we present two experiments in which we compare our method with the method proposed by Boykov and Kolmogorov (BK) [3]. The first experiment measures the metrication error of both approximations for lines under different angular orientations. Assuming a straight line under all possible angular orientations we compare its length in the cut metric (i.e. the sum of the weights of the edges it severs) to the ground-truth (i.e. the actual Riemannian

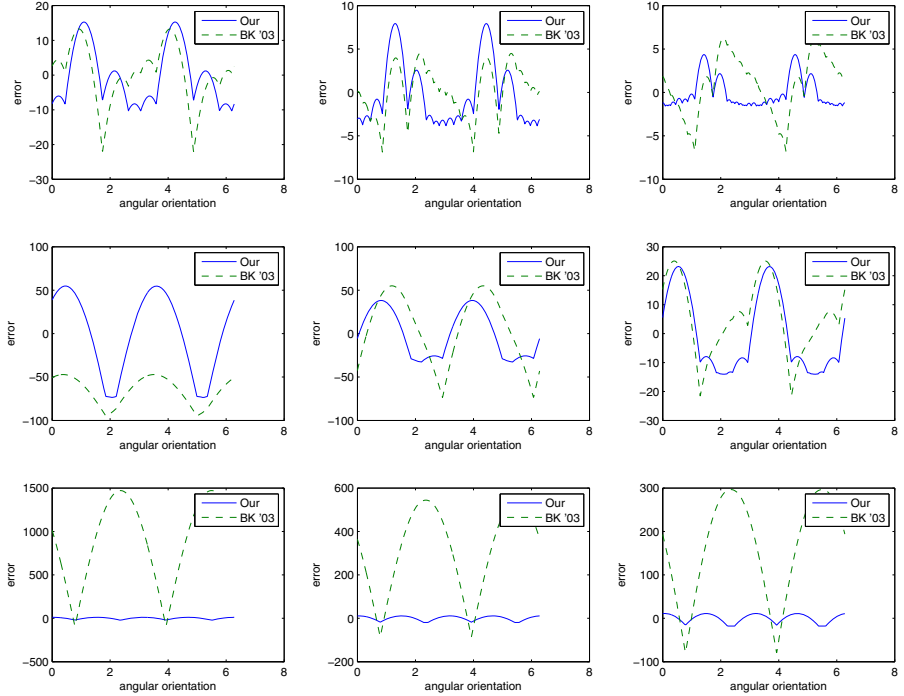


Fig. 3. Multiplicative metrication error (in percents) of the approximation for various metric tensors. Comparison of our and Boykov and Kolmogorov’s [3] method. Left column: \mathcal{N}_8 . Middle column: \mathcal{N}_{16} . Right column: \mathcal{N}_{32} . Top row: Space dilation under 10° by a factor of 2. Middle row: Space dilation under 60° by a factor of 15. Bottom row: Space dilation under 45° by a factor of 40.

length of the line) for several combinations of neighbourhood and a metric tensor. The multiplicative error of both approximations is depicted in Fig. 3. For small stretch factors (first two rows) the performance of both approaches is similar. Nevertheless, while the error of our method oscillates around zero in all cases, the BK method behaves unexpectedly in some situations underestimating the length under all angular orientations (first graph on the second row). With growing stretch factor the maximal error grows for both methods, particularly due to the growing value of $\sup_k \Delta\phi_k$ (noticeable in Fig. 2b). However, for larger stretch factors there is a huge difference in favor of our method. Especially for small neighbourhoods the assumption of infinitely small $\Delta\phi_k$ required to derive Eq. 7 is unrealistic causing a considerable error. Unfortunately, in practice even much larger stretch factors are common in which case the performance of both considered methods is going to be relatively poor unless a very dense neighbourhood is used.

In the second experiment we have conducted a comparison on a practical image segmentation problem using the geodesic segmentation model [73]. We have used a 16-neighbourhood and the same parameters for both methods. For

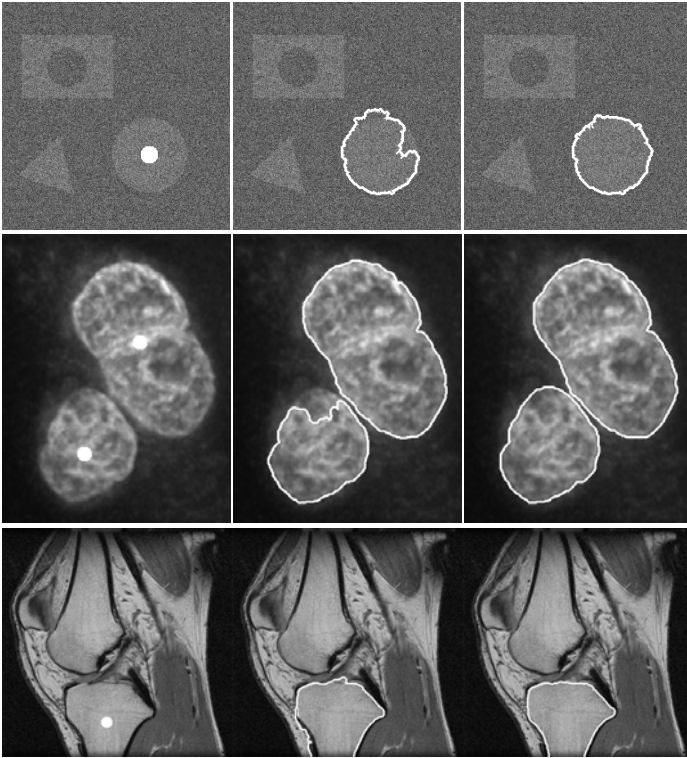


Fig. 4. Graph cut based geodesic image segmentation. Left column: Input image – synthetic and biomedical data. White foreground marker is placed inside the object(s) to be segmented. Middle column: Segmentation result using the Riemannian metric approximation of [3]. Right column: Result using the proposed method. 16-neighbourhood and the same parameters were used for both methods.

each of the images depicted in Fig. 4 a foreground marker was placed inside the objects to be segmented with background marker being the border of the image. Subsequently, image gradient was calculated in each pixel and a metric tensor was constructed dilating the space in the direction of the gradient. Exact formula for construction of the matrix M is available in [3]. Finally, a geodesic in the Riemannian space was found separating the foreground and background markers using the graph cut algorithm yielding the segmentation boundary. Apparently, our approach to the Riemannian metric approximation gives significantly better results for the particular examples. Besides these experiments, more practical tests were performed and, generally speaking, we have not found any example in which our method gives inferior results.

5 Discussion

In this section we would like to focus on the computational demands of the proposed method. As we have already demonstrated we are able to achieve smaller

approximation errors, however, this comes for a price. While the BK method consisting of Eq. 7 has generally constant complexity (few basic mathematical operations) our method is clearly more demanding requiring the computation of circular or spherical Voronoi diagram in 2D or 3D, respectively, to obtain $\Delta\phi_k^v$. While this does not present a computational burden in 2D (it is equally fast when the neighbourhood vectors are sorted according to their angular orientation beforehand) it may cause a significant computational overhead in 3D. For instance, in cases where the metric tensor differs in each pixel, the spherical Voronoi diagram has to be recalculated repeatedly which results in our method being approximately 20 times slower making it hardly feasible. However, it is still useful in case of scalar (i.e., isotropic) or constant (such as those simulating anisotropic resolution in 2D and 3D [9]) Riemannian metrics where it is enough to compute the Voronoi diagram only once. Then again the method is equally fast as the BK approach. Also note that the difference between the two approximations can be sometimes minor in practice. Thus, a decision has to be always made regarding the trade-off between the precision and speed.

Another complication in our method that can be encountered is the construction of the transformation matrix T . In general, the construction of T from the matrix M requires spectral decomposition of M which can be slow. However, this can be avoided in most situations by directly constructing the matrix T instead of M such as in the image segmentation example where the eigenvectors (the image gradient) and eigenvalues (computed from the length of the gradient) are known so T can be constructed directly.

6 Conclusion

In this paper, we have presented a novel method for Riemannian metric approximation via graph cuts. It is based on the Cauchy-Crofton formula and is a generalization of our previously devised formulas for the Euclidean metric approximation on anisotropic grids. Using a transformation matrix we reduce the Riemannian case to the Euclidean one and modify our edge weight formulas to be able to cope with non-orthogonal grids. The proposed approximation has a smaller metrication error than other state-of-the-art methods which was verified both in theoretical and practical experiments. Implementation of the method described in this paper can be downloaded from <http://cbia.fi.muni.cz/projects/graph-cut-library.html>.

Acknowledgment. This work has been supported by the Ministry of Education of the Czech Republic (Projects No. MSM-0021622419, No. LC535 and No. 2B06052).

References

1. Appleton, B., Talbot, H.: Globally minimal surfaces by continuous maximal flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28, 106–118 (2006)
2. Boykov, Y., Funka-Lea, G.: Graph cuts and efficient n-d image segmentation (review). *International Journal of Computer Vision* 70(2), 109–131 (2006)

3. Boykov, Y., Kolmogorov, V.: Computing geodesics and minimal surfaces via graph cuts. In: ICCV 2003: Proceedings of the Ninth IEEE International Conference on Computer Vision, p. 26 (2003)
4. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26(9), 1124–1137 (2004)
5. Boykov, Y., Veksler, O.: Graph Cuts in Vision and Graphics: Theories and Applications. In: *Handbook of Mathematical Models in Computer Vision*, pp. 79–96. Springer, Heidelberg (2006)
6. Brown, K.Q.: Geometric transforms for fast geometric algorithms. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, USA (1979)
7. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. In: *IEEE International Conference on Computer Vision*, pp. 694–699 (1995)
8. Chan, T.F., Vese, L.A.: Active contours without edges. *IEEE Transactions on Image Processing* 10(2), 266–277 (2001)
9. Daněk, O., Matula, P.: Graph cuts and approximation of the euclidean metric on anisotropic grids. In: *VISAPP 2010: International Conference on Computer Vision Theory and Applications*, vol. 2, pp. 68–73 (2010)
10. Kolmogorov, V., Boykov, Y.: What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. In: *ICCV 2005: Proceedings of the Tenth IEEE International Conference on Computer Vision*, pp. 564–571 (2005)
11. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26(2), 147–159 (2004)
12. Osher, S., Sethian, J.A.: Fronts propagating with curvature dependent speed: Algorithms based on Hamilton–Jacobi formulation. *Journal of Computational Physics* 79(1), 12–49 (1988)
13. Strang, G.: Maximum flows and minimum cuts in the plane. *Journal of Global Optimization* 47, 527–535 (2010)
14. Vogiatzis, G., Torr, P.H.S., Cipolla, R.: Multi-view stereo via volumetric graph-cuts. In: *CVPR 2005: IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 391–398 (June 2005)
15. Zach, C., Niethammer, M., Frahm, J.M.: Continuous maximal flows and wulff shapes: Application to mrfs. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1911–1918 (2009)
16. Zeng, Y., Chen, W., Peng, Q.: Efficiently solving the piecewise constant mumford-shah model using graph cuts. Tech. rep., Dept. of Computer Science, Zhejiang University, P.R. China (2006)

Introduction to Digital Level Layers

Yan Gérard, Laurent Provot, and Fabien Feschet

Univ. Clermont 1, ISIT, Campus des Cézeaux, 63172 Aubière, France
yan.gerard@u-clermont1.fr, provot.research@gmail.com, feschet@acm.org

Abstract. We introduce the notion of Digital Level Layer, namely the subsets of \mathbb{Z}^d characterized by double-inequalities $h_1 \preceq f(x) \preceq h_2$. The purpose of the paper is first to investigate some theoretical properties of this class of digital primitives according to topological and morphological criteria. The second task is to show that even if we consider functions f of high degree, the computations on Digital Level Layers, for instance the computation of a DLL containing an input set of points, remain linear. It makes this notion suitable for applications, for instance to provide analytical characterizations of digital shapes.

Keywords: Digital Primitives, Cover, Thickness, Linear Programming, Support Vector Machines.

1 About the Words

We propose to introduce in this paper the notion of Digital Level Layers. This name refers to level sets, namely to subsets of a space X characterized by an equation $f(x) = h$ where the variable x is in X and where h is a constant value. Most often the space X is \mathbb{R}^d . Under assumption that the gradient of f is not null, level sets are sub-manifolds of dimension $d-1$, namely hypersurfaces. In this paper we are interested in digital sets, which means that the space X is restricted from \mathbb{R}^d to \mathbb{Z}^d . In order to preserve interesting properties, this restriction of support space requires to relax the constraint $f(x) = h$ to a double-inequality $h_1 \leq f(x) \leq h_2$. It leads to the definition of Digital Level Layers: *Digital* because the space X is \mathbb{Z}^d , *Level* because we use a height function f and *Layers* because we consider the layer between the two level sets $f(x) = h_1$ and $f(x) = h_2$.

Definition 1. A Digital Level Layer –DLL for short– is a subset of points x of \mathbb{Z}^d verifying a double inequality $h_1 \preceq f(x) \preceq h_2$ where f goes from \mathbb{Z}^d to \mathbb{R} , where the symbols \preceq and \preccurlyeq denote \leq or $<$ and where h_1, h_2 are real numbers.

2 Why This Kind of Primitives ?

One of the main challenges of digital geometry is to provide digital primitives corresponding to algebraic curves or surfaces in the Euclidean world. There exist several ways to define such objects. The first idea to introduce geometrical digital primitives is to consider the trace on \mathbb{Z}^d of the geometrical primitives of \mathbb{R}^d . It means that a digital straight line would be the trace of a real straight line, a

digital circle the trace of a real circle, namely the set of solutions of multivariate Diophantine equations. Although the resolution of such equations is undecidable, the main reason why they are not used for geometrical purposes is that their visual rendering is completely different from the visual rendering of the Euclidean objects – in fact they have bad properties in terms of digital topology: they have not enough voxels to separate the space, which can be traduced by tunnels. Since this first naive idea is not satisfactory, one can wonder how to define digital primitives that fulfill some good topological and morphological properties. Several answers to this kind of problem have been developed for the last thirty years. We can mention two kinds of approaches: “morphological” and “topological”.

- the morphological approach is to consider an extended notion of trace. A point $x \in \mathbb{Z}^d$ belongs to the trace of the Euclidean surface $S \subset \mathbb{R}^d$ if the hypercube $[x - \epsilon, x + \epsilon]^d$ has a non-empty intersection with S . This extended notion of trace is equivalent to the trace of the Minkowski sum $S + [x - \epsilon, x + \epsilon]^d$. These objects are simply called “covers” [6] (Fig. 1) and with some refinements “supercovers” [5,2] (*supercover* deals with the cases where, if $\epsilon \geq 1/2$, a real point of S can provide several integer points and hence a non minimal thickness). We can also use different “structuring elements” such as Euclidean balls, hypercubes, segments... With the square $[-0.5; 0.5]^2$, the cover of $S + [-0.5; 0.5]^2$ corresponds to the set of pixels centered on the grid points that cross the circle. If we choose a segment $[-0.5; 0.5] \times \{0\}$ or $\{0\} \times [-0.5; 0.5]$ (depending on the region) as structuring element, we have another definition of circle which is also widely used. Generally speaking, it can be hard to provide a general analytic characterization of these primitives but as we will notice in the sequel, local characterizations can be found.
- a topological approach can be used if the Euclidean primitive we want to digitize has an interior and an exterior. Hence a natural way to define the digitization of an implicit curve or surface $f(x) = h$ is for instance to consider the boundary of the trace of the interior integer points ($f(x) < h$). The main advantage of this topological approach is to define objects which are topologically guaranteed (Fig. 1).

Hence is there a reason to introduce another class of primitives? In fact each one of these two classes of digital surfaces has its own drawbacks. The main problem with morphological surfaces is their recognition. As far as we know, no efficient algorithm has been published to recognize ellipsoids for instance, defined according to this approach. The topological surfaces do not suffer from the same problem. Their recognition is a problem of separation. To recognize for instance an ellipsoid, the problem is to find a Euclidean ellipsoid separating the interior points from their complement. This problem is solved by Support Vector Machines [7,15]. There is however another reproach we can do for both approaches: they provide no direct analytical characterization of the voxels of the surface. The main interest of Digital Level Layers is to satisfy this requirement which can be useful in the framework of conversions from raster to vector graphics. In other words, DLL is the third approach, the “analytical” one, to define digital primitives.

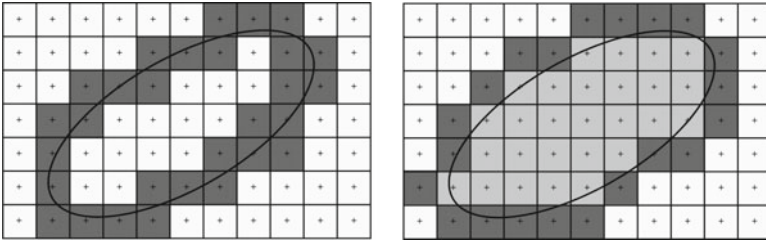


Fig. 1. Digitization's of an ellipse, on the left according to the morphological principle – we take the pixels which cross the Euclidean ellipse – on the right according to the topological principle – we take the difference between the pixels that center is in the interior of the ellipse and their 4-neighborhood

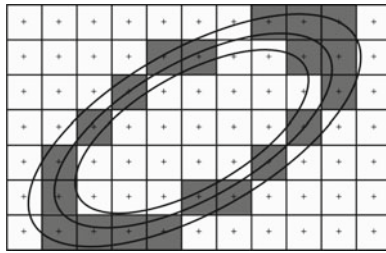


Fig. 2. Digitization of an ellipse as a Digital Level Layer – we take the integer points between two level sets on both sides of the ellipse

The main drawback of DLL as “surfaces” appears on Fig. 2: the width of DLL is not constant all along its points. An idea could be to say: “OK, instead of using a double inequality which can be written $0 < f(x) < h$ where h is a constant, we just have to work with inequalities $0 < f(x) < h(x)$ where $h(x)$ depends linearly on the Euclidean norm of the gradient $\vec{\nabla} f$ namely $h(x) = h_0 \|\vec{\nabla} f(x)\|$ with a constant h_0 ” but the inequality $0 < f(x) < h_0 \|\vec{\nabla} f(x)\|$ can be simply rewritten $0 < \frac{f(x)}{\|\vec{\nabla} f(x)\|} < h_0$. It is just a change of function f and the gradient of the new function $\frac{f(x)}{\|\vec{\nabla} f(x)\|}$ is not guaranteed to have a constant norm: the problem remains!

Is it a sufficient reason to reject DLL? No because in spite of its main drawback, there are a lot of situations where we do not take care of the Euclidean width of the surface. The main question is thus to determine whether Digital Level Layers can be useful. The outline of the paper is to investigate the interest of this notion through two main angles:

- from a theoretical point of view: which theoretical properties can we obtain, with some assumptions on the function f , for instance if f is a polynomial of bounded degree? We are going to consider topological and morphological properties.

- the second important point is to determine the algorithmic tools that can deal with DLL. We focus our attention on the recognition of these families of primitives – with again some assumptions on f . The recognition problem has been deeply investigated in the framework of linear structures and we provide here a new approach based on excluded sets of points. We are going to see how the results obtained in a linear framework can be extended in order to work efficiently with DLL.

3 Theory

The definition of Digital Level Layers is very general, maybe too much, as soon as we make no assumption on the function f . We can even notice that for any finite subset S of \mathbb{Z}^d , we can construct the analytic expression of a function f such that S is the DLL of double-inequality $-1/2 \leq f(x) \leq 1/2$. Hence to say that a given set is a DLL has clearly no interest by itself. The interest of the notion is in the relation between the set S , the class of the function f that appears in the double-inequality $h_1 \preceq f(x) \preceq h_2$ and the value of the difference $h_2 - h_1$.

3.1 Functional and Algebraic DLL

We start by introducing the notion of *functional* DLL.

Definition 2. A Digital Level Layer L is called *functional in direction i* if it contains exactly one point in each line of the lattice \mathbb{Z}^d parallel to the i^{th} axis.

We call these DLL functional because they define clearly a function from \mathbb{Z}^{d-1} to \mathbb{Z} and conversely any function from \mathbb{Z}^{d-1} to \mathbb{Z} can be represented by a functional DLL. Most often, functional DLL are used to digitize real functions f sending \mathbb{Z}^{d-1} or even \mathbb{R}^{d-1} to \mathbb{R} by taking their integer part $\lfloor f \rfloor$ (other convention can be chosen as $\lfloor f + 0.5 \rfloor$, $\lceil f + 0.5 \rceil$ or $\lceil f \rceil$). The functional DLL associated to $\lfloor f \rfloor$ is simply characterized by double-inequality $0 \leq f(x_i)_{1 \leq i \leq d-1} - x_d < 1$. Such functional DLL are also morphological surfaces since we can obtain them as the trace on \mathbb{Z}^d of the sum of the surface $x_d = f(x_i)_{1 \leq i \leq d-1}$ with the vertical segment $] - 1; 0]$ as structuring element.

As we will see in the section devoted to algorithms, digital hyperplanes are important for the recognition of DLL and they are particular cases of these generic class of objects: A digital straight line is a DLL with $f(x) = a_1x_1 + a_2x_2$, a digital plane is a DLL with $f(x) = a_1x_1 + a_2x_2 + a_3x_3$ and more generally digital hyperplanes are simply defined with an affine form $f(x) = \sum_{i=1}^d a_i x_i$ as height function [13]. DLL cover also the case of digital hyperspheres $(R - \frac{1}{2})^2 \leq \sum_{i=1}^d (x_i - a_i)^2 < (R + \frac{1}{2})^2$ (according their definition in [4]) (Fig. 4). All these usual primitives of digital geometry are DLL with a polynomial height function f of low degree. It leads to the introduction of the notions of *algebraic* DLL (Fig. 5).

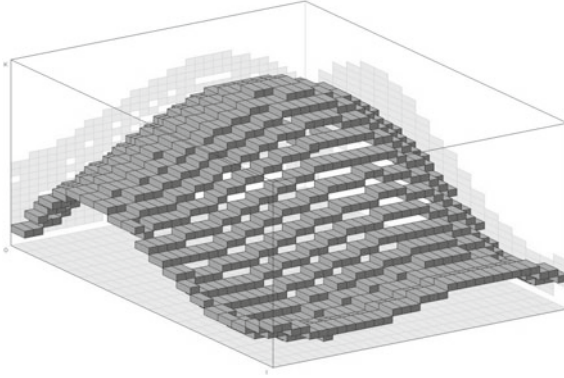


Fig. 3. The functional DLL of double-inequality $4680370559422800 \leq -17758960839495486x + -2015536714762185y + 292165823537172xy + 2304539315667564x^2 + -172062298114380y^2 + -8936152895940xy^2 + -75980632598934x^3 + 1625988388601x^2y + 7065487923165y^3 + 4336388981634600z < 4680370559422800 + 4336388981634600$ namely $z = \lfloor (4680370559422800 + 17758960839495486x + 2015536714762185y - 292165823537172xy - 2304539315667564x^2 + 172062298114380y^2 + 8936152895940xy^2 + 75980632598934x^3 - 1625988388601x^2y^1 - 7065487923165y^3)/4336388981634600 \rfloor$

Definition 3. An algebraic Digital Level Layer is a subset of points x of \mathbb{Z}^d verifying a double inequality $h_1 \preccurlyeq P(x) \preccurlyeq h_2$ where P is a multivariate polynomial of $\mathbb{R}[X_i]_{1 \leq i \leq d}$ and where h_1, h_2 are real numbers.

It follows of course that digital lines, hyperplanes and spheres (for at least one definition) are algebraic DLL.

3.2 Topological Properties

We consider in this section a notion of digital connectedness induced by a norm N of \mathbb{R}^d (two points x and x' of \mathbb{Z}^d are neighbors if their distance $N(x' - x) \leq 1$) or equivalently by a convex neighborhood which can be translated in any point of the lattice \mathbb{Z}^d . The ball of radius 1 is denoted B_1 .

We recall that the dual norm N^* of N is defined by its ball of radius 1 with $B_1^* = \{x \in \mathbb{R}^d / \forall y \in B_1, x \cdot y \leq 1\}$. The balls B_1 and B_1^* are polar from each other [16]. We have for instance $N_\infty^* = N_1$ and conversely $N_1^* = N_\infty$. The connectedness associated to N_∞ and N_1 is respectively denoted $3^d - 1$ and $2d$ -connectedness. The duality of N_∞ and N_1 leads of course to the idea that the $3^d - 1$ and the $2d$ -connectedness from \mathbb{Z}^d are dual.

Connectedness. By considering a digital layer L_δ of double inequality $h - \frac{\delta}{2} \leq f(x) < h + \frac{\delta}{2}$, three domains are considered:

- the two domains of the complement of L_δ in \mathbb{Z}^d denoted L_δ^- and L_δ^+ and verifying respectively $f(x) < h - \frac{\delta}{2}$ and $f(x) \geq h + \frac{\delta}{2}$
- the layer L_δ itself.

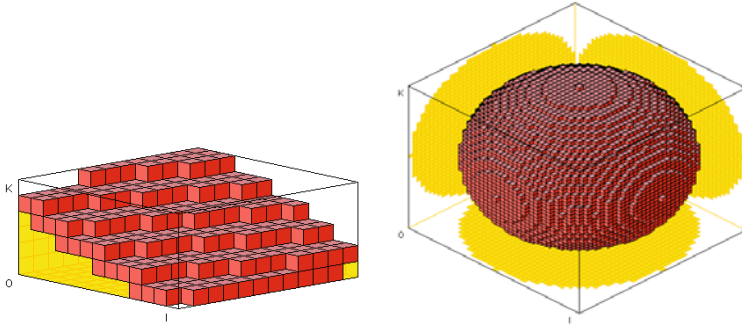


Fig. 4. The digital plane of double-inequality $583 \leq 72x - 28y + 190z \leq 845$ and the digital sphere of double-inequality $19^2 < (x - 20)^2 + (y - 20)^2 + (z - 20)^2 \leq 20^2$

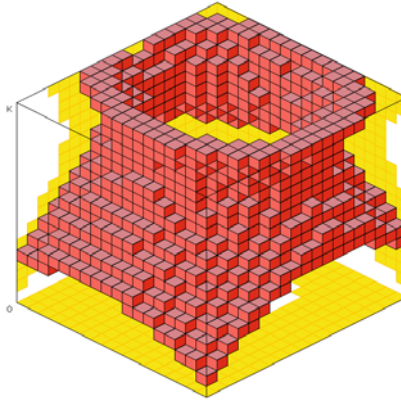


Fig. 5. The algebraic DLL of double-inequality $-233050932002 \leq 9706766352x + 34545685472y - 50669609424z - 1006860048x^2 - 1419723424y^2 + 1813925040z^2 + 346913568xz - 191023296yz \leq -194272584000$

We notice that by increasing the value δ , points are deleted from L^- and L^+ to be added in L . We introduce the set $\Delta = \{\delta \in \mathbb{R}^+ / L_\delta \text{ is connected}\}$. As in a Euclidean framework, the property of Δ to be an interval is related to the existence of local minimum of $|f - h|$ upper than the smaller value δ which provides a connected layer L_δ .

To determine the set Δ is often difficult. Except in the case of the digital straight lines of \mathbb{Z}^2 , namely with functions $f(x) = a_1x_1 + a_2x_2$ (we assume $GCD(a_1, a_2) = 1$) where it is known for a long time that $\Delta = [|a_1| + |a_2|, +\infty[$ according to 4-connectedness and $\Delta = [\max\{|a_1| + |a_2|\}, +\infty[$ according to 8-connectedness, the problem has been longly investigated. In dimension 3 with digital planes of \mathbb{Z}^3 having a rational direction, Δ remains an interval of the form $\Delta = [\min, +\infty[$ that minimal connecting thickness has been expressed in [11]. In the case of digital planes of given irrational normal (a_1, a_2 and a_3 are rationally independent), the lower bound of the connecting thickness has been computed in 2009 in [8].

Separation. With the question of connectedness of L_δ arises also the question of the connectedness between L_δ^- and L_δ^+ , namely the existence of tunnels in L_δ .

Proposition 1. *We assume that f is C^1 and that at least one of the two symbols \preceq or \preceq' of inequality $h - \delta/2 \preceq f(x) \preceq' h + \delta/2$ is strict ($<$). If for every point $x \in \mathbb{R}^d$ we have $N^*(\vec{\nabla} f(x)) \leq \delta$, then L_δ^- and L_δ^+ are separated according to the connectedness associated to the norm N (L_δ is tunnel-free).*

Proof. We assume that for any x in \mathbb{R}^d we have $N^*(\vec{\nabla} f(x)) < \delta$ and that the two points x and x' are adjacent according to N namely that $N(x' - x) \leq 1$. We have $f(x') - f(x) = \int_x^{x'} \vec{\nabla} f(x) \cdot \frac{(x' - x)}{\|x' - x\|_2} ds$ where the path of integration is the segment $[x, x']$. It follows from $N^*(\vec{\nabla} f(x)) \leq \delta$ that $\vec{\nabla} f(x) \cdot (x' - x) \leq \delta N(x' - x)$ or again $\vec{\nabla} f(x) \cdot \frac{x' - x}{\|x' - x\|_2} \leq \delta \frac{N(x' - x)}{\|x' - x\|_2}$. Thus we have $f(x') - f(x) \leq \int_x^{x'} \delta \frac{N(x' - x)}{\|x' - x\|_2} ds$. We obtain $f(x') - f(x) \leq \delta N(x' - x) \leq \delta$. Hence, x and x' cannot be one in L_δ^- and the other in L_δ^+ .

Proposition [1](#) means that a sufficient condition of separation for the digital line of double-inequality $h - \delta/2 \leq a_1x + a_2y < h + \delta/2$ is $N^*(\vec{\nabla} f(x)) \leq \delta$. We have of course $\vec{\nabla} f(x) = (a_1, a_2)$. It means that a sufficient condition for separability is $N_1^*(a_1; a_2) \leq \delta$ namely $N_\infty(a_1; a_2) \leq \delta$ i.e. $\max(|a_1|; |a_2|) \leq \delta$ for 4-separability. It means that $N_\infty^*(a_1; a_2) \leq \delta$ namely $N_1(a_1; a_2) \leq \delta$ i.e. $|a_1| + |a_2| \leq \delta$ is sufficient to guarantee 8-separability. These two conditions are sufficient to have a tunnel-free line and it is known that they are also necessary [3](#).

It is the same in the framework of digital planes. The sufficient condition of separability given by Proposition [1](#) is $\max(|a_1|; |a_2|; |a_3|) \leq \delta$ for 6-connectedness while it is $|a_1| + |a_2| + |a_3| \leq \delta$ for 26-connectedness. In the general framework of digital hyperplanes (double inequality $h - \delta/2 \leq a \cdot x < h + \delta/2$), the Proposition [1](#) proves the sufficient condition of separability $N_\infty(a) \leq \delta$ for the $2d$ connectedness and $N_1(a) \leq \delta$ for the $3^d - 1$ connectedness. This condition is also necessary [3](#).

3.3 Morphological Property

The main morphological property of surfaces is their thinness. As noticed previously, the main drawback of DLL is to provide a representation of a Euclidean surface $f(x) = h$ which can be thick in some points and thin elsewhere. It is however not the case of all digital layers, as we can see with digital lines or digital spheres and the fact that some layers are not satisfactory due to their variation of thickness does not mean that it is the case for all DLL.

One of the tools we provide in the next section is precisely a method to compute layers that thickness is locally controlled by forbidden points (which can be chosen arbitrarily or automatically). Even if the layer obtained this way has a variable thickness somewhere, we believe that its local control – with some “control” points – in a region of interest makes DLL an interesting tool for digital geometry applications, and a step towards an efficient vectorization of digital shapes.

3.4 Comparison between DLL and Morphological Surfaces

Morphological surfaces – covers – are defined as the trace on \mathbb{Z}^d of the Minkowski’s sum $S + E$ of a Euclidean surface S with a structuring element E which can be a unit vertical segment $0^{d-1} \times [-0.5; 0.5[$, a unit voxel $[-0.5; 0.5]^d$ or any convex subset of \mathbb{R}^d . The boundary of the sum $S + E$ is a subset of the sum of boundaries $\partial S + \partial E$. Its structure is obtained by translating alternatively some pieces of the boundary of S and some other pieces of the boundary of E . An example is drawn in Fig. 6. We can decompose the ellipse of equation $f(x) = h$ in four pieces according to the quadrant of the normal vector $\overline{\nabla} f$. A pixel of center x crosses the ellipse if and only if it belongs to a set of equation $f(x - v) \leq h \leq f(x + v)$ where v can be $(-0.5; -0.5)$, $(-0.5; 0.5)$, $(0.5; -0.5)$ or $(0.5; 0.5)$ or to a unit square centered on the four points of extremal coordinates. Without providing the details, it means that the morphological ellipse can be decomposed in four squares and four pieces of curved stripes characterized by inequalities $f(x - v) \leq h \leq f(x + v)$. We can notice that such double-inequality is completely different from the double-inequalities characterizing the DLL. We have here two different functions $f(x - v)$ and $f(x + v)$ translated one from each other, while in DLL we have only one function and it is the constant that differs to define the two bounding surfaces. It follows that DLL are deeply different from morphological surfaces.

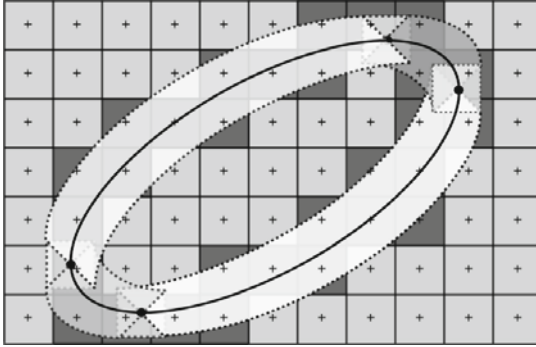


Fig. 6. The cover of an ellipse namely the digital morphological ellipse obtained with a unit square as structuring element. It is made of the integer points belonging to several pieces obtained by translating the initial curve from $(-0.5; -0.5)$, $(-0.5; 0.5)$, $(0.5; -0.5)$, $(0.5; 0.5)$ and to unit squares centered at the extremal points.

4 Algorithms

The main algorithmic challenge with digital primitives is their recognition and the main advantage of DLL on morphological surfaces is that this problem can be tackled efficiently. The important fact is that even with algebraic DLL of degree greater than 1, the recognition problem remains linear. This reduction is known as the “kernel trick”. It has been introduced in 1964 in [1]. Hence

we decompose the section into two parts. The first one is devoted to problems of “linear” computational geometry while the second one is devoted to their application in the framework of general DLL.

4.1 Two Problems of Computational Geometry

Find the thinnest strip containing a given set of points The first problem that we consider is the following:

Problem 1. Input: A finite subset In of \mathbb{R}^d . *Output:* Find the thinnest affine strip $h_1 \leq n \cdot x \leq h_2$ containing In (where $n \cdot x$ denotes the dot product between the normal vector n and the variable point x).

We should precise what we mean by “thin”. A natural choice is to give to the word thin an Euclidean meaning: The Euclidean thickness of the affine strip $h_1 \leq n \cdot x \leq h_2$ is the ratio between $h_2 - h_1$ and the Euclidean norm of the normal vector $\|n\|_2$ but we could of course define the thickness with any other norm. It provides a definition of thickness for any subset of \mathbb{R}^d :

Definition 4. *The thickness of a subset In of \mathbb{R}^d according to a norm $\|\cdot\|$ is the lower bound of the set of thickness of the strips containing In .*

We can even extend the notion of thickness from norms to functions which are just homogeneous. Instead of considering that the thickness of the strip $h_1 \leq n \cdot x \leq h_2$ is the ratio $\frac{h_2 - h_1}{\|n\|}$, it becomes $\frac{h_2 - h_1}{\varphi(n)}$ where φ is a homogeneous function i.e satisfying $\varphi(\lambda n) = \lambda \varphi(n)$. This extension allows us to speak about directional thickness if we consider for instance $\varphi(n) = |n_d|$ where n_d is the last coordinate of n .

Definition 5. *The φ -thickness of a subset In of \mathbb{R}^d is the lower bound of the set of values $\frac{h_2 - h_1}{\varphi(n)}$ where the strip $h_1 \leq n \cdot x \leq h_2$ contains In .*

In the case of a discrete set In , the minimum may be reached for some strips. The question is: given φ , how do we solve Problem [11](#)? For the Euclidean norm ($\varphi(\cdot) = \|\cdot\|_2$), the solution can be computed by the rotating caliper [14](#) based on the computation of the convex hull. For a directional width ($\varphi(\cdot)$ is the absolute value of a linear form), the problem can be expressed as a linear program (minimize δ subject to $h \leq \sum_{i=1}^{d-1} n_i x_i + x_d \leq h + \delta$) and solved in linear time (if d is fixed) by Megiddo algorithm [12](#). We can also solve it efficiently with the chord algorithm [9](#) or others. If $\varphi(\cdot)$ is a polyhedral norm, namely a norm whose unit ball is a polyhedron, we can compute the φ -thickness as the minimum of all the directional thickness in the directions normal to the faces of the unit polyhedron. We should however notice that in the framework of digital hyperplanes recognition, the problem arises under a different form. Instead of computing the minimal φ -thickness of In , the question is to determine whether this value is smaller than a given maximal thickness:

Problem 2. Input: A finite subset In of \mathbb{R}^d , a homogeneous function $\varphi : \mathbb{R}^d \rightarrow \mathbb{R}$ and a maximal thickness M . *Output:* Find a strip $h_1 \leq n \cdot x \leq h_2$ containing In with a φ -thickness $< M$.

The answer of Problem 1 provides clearly the answer of Problem 2, and usually the algorithms chosen to compute the φ -thickness can be modified so that they stop as soon as the φ -thickness of In becomes greater than M . As known in the framework of Linear Programming, there is usually no great difference in the algorithms computing the minimum of an objective function and the ones determining whether the minimum is greater than a constant value.

Find a strip containing a set of points with two excluded sets of points.

Let us now consider a different problem without using any notion of thickness. It is just based on two sets of forbidden points Up and $Down$. The aim is to put the initial set In in a strip, Up on one side and $Down$ on the other side.

Problem 3. Input: Three finite subsets $In, Up, Down$ of \mathbb{R}^d . *Output:* Find an affine strip $h_1 \leq n.x \leq h_2$ containing In with the points of Up and $Down$ on both sides namely with $h_1 \leq n.x \leq h_2$ for all x in In , with $n.x < h_1$ for $x \in Down$ and with $n.x > h_2$ for x in Up .

Here we have a problem of multiclass separation. These problems have been investigated in the deep SVM literature [7,15] and we can solve Problem 3 efficiently with a variant of the well-known GJK algorithm [10] used for collision detection.

4.2 From Affine Strips to DLL Recognition: The Kernel Trick

In the case of digital straight lines, planes and hyperplanes, the problem of recognition is to determine whether a given set of points In belongs to a digital hyperplane with a fixed maximal thickness. In the extended framework of DLL, the problem arises in the same way. Let us consider a DLL characterized by $h_1 \leq f(x) < h_2$ where the function $f(\cdot) : \mathbb{Z}^d \rightarrow \mathbb{R}$ belongs to the linear space F generated by functions $f_1(\cdot), f_2(\cdot), \dots, f_m(\cdot)$. We define the thickness of the double-inequality $h_1 \leq f(x) < h_2$ from any homogeneous function $\varphi : F \rightarrow \mathbb{R}$.

Definition 6. *The φ -thickness of double inequality $h_1 \leq f(x) \leq h_2$ where f is in the linear space F with a homogeneous real function $\varphi : F \rightarrow \mathbb{R}$ is $\frac{h_2-h_1}{\varphi(f)}$.*

The main drawback of this notion of φ -thickness is to be algebraic. It can have a geometrical meaning, mainly as a directional thickness, to compute for instance functional DLL as in Fig. 3 but in many other cases, this notion of thickness remains mainly algebraic without any possibility to have a direct geometrical understanding of its value. We can however be interested in recognizing thin DLL, at least for the cases where a geometrical meaning holds.

Problem 4. Input: A finite subset In of \mathbb{R}^d , a linear space F of functions $f : \mathbb{Z}^d \rightarrow \mathbb{R}$ generated linearly by $f_1(\cdot), f_2(\cdot), \dots, f_m(\cdot)$, a homogeneous function $\varphi : F \rightarrow \mathbb{R}$ and a maximal thickness M . *Output:* Find a DLL containing In having a double-inequality $h_1 \leq f(x) \leq h_2$ with a φ -thickness $< M$.

This problem consists in finding h_1, h_2 , and coefficients a_1, a_2, \dots, a_m of $f = \sum_{i=1}^m a_i f_i$ with $\frac{h_2-h_1}{\varphi(\sum_{i=1}^m a_i f_i)} < M$ under constraints $h_1 \leq \sum_{i=1}^m a_i f_i(x) \leq h_2$ for

all x in In . If we denote $g(x) = (f_i(x))_{1 \leq i \leq m} \in \mathbb{R}^m$ and n the vector $(a_i)_{1 \leq i \leq m}$, the problem is reduced to solve Problem 2 with the set $g(In)$ as input (in dimension m) and function $\varphi(n) = \varphi(\sum_{i=1}^m a_i f_i)$ as thickness criterion. We can solve it with the rotating caliper if φ is a quadratic norm and by linear programming or chords algorithm if φ is the absolute value of a linear form or a polyhedral norm. As corollary, these tools allow to solve efficiently Problem 4. Such algorithms have been implemented successfully for the directional thickness: Fig. 3 and Fig. 5 have been for instance obtained by using these tools.

As noticed previously, the computation of a DLL of minimal φ -thickness or of φ -thickness $< M$ does not guarantee to have a DLL with a minimal local geometrical thickness. That is the reason why we have introduced Problem 3. Instead of providing the set In as the only input set and to look for a DLL of minimal algebraic thickness containing it – since the notion of φ -thickness has not always the geometrical meaning that we can hope – we suggest to replace it by outliers. The idea is to provide two control sets Up and $Down$ whose points should be bypassed by the DLL. With the kernel trick, we can reduce the computation of a DLL containing In , with the outliers of $Down$ and Up on one and the other side, to Problem 3 with $g(In)$, $g(Up)$ and $g(Down)$ as input sets. A variant of the GJK algorithm can be used to solve it efficiently. This alternative to the recognition of a DLL of minimal thickness allows us to imagine many effective applications and can be particularly interesting to go from raster graphics to vector graphics without dealing with an unsuitable notion of algebraic thickness.

5 Conclusion

From the theoretical point of view, it is rather clear that Digital Level Layers do not have the nice arithmetical properties of digital straight lines or planes. We also have emphasized the main drawback of DLL compared to morphological and topological surfaces: they have a non constant local geometrical thickness. The lack of geometrical meaning of their algebraic thickness makes DLL recognition under the classical approach (Problem 4) not really suitable for applications. It is much more interesting in practice to consider the recognition of DLL by replacing this condition of maximal algebraic thickness by two sets of outliers. With this new approach, which can be solved very efficiently (Problem 3), Digital Level Layers become a promising tool in the framework of vectorization of raster graphics.

References

1. Aizerman, M., Braverman, E., Rozonoer, L.: Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control* 25, 821–837 (1964)
2. Andres, E.: The supercover of an m -flat is a discrete analytical object. *Theoretical Computer Science* 406(1-2), 8–14 (2008)

3. Andres, E., Acharya, R., Sibata, C.: Discrete analytical hyperplanes. *Graphical Model and Image Processing* 59(5), 302–309 (1997)
4. Andres, E., Jacob, M.A.: The discrete analytical hyperspheres. *IEEE Transactions on Visualization and Computer Graphics* 3(1), 75–86 (1997)
5. Brimkov, V.E., Andres, E., Barneva, R.P.: Object discretizations in higher dimensions. *Pattern Recognition Letters* 23(6), 623–636 (2002)
6. Cohen-Or, D., Kaufman, A.E.: Fundamentals of surface voxelization. *Graphical Model and Image Processing* 57(6), 453–461 (1995)
7. Cristianini, N., Shawe-Taylor, J.: *Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, Cambridge (2000)
8. Domenjoud, E., Jamet, D., Toutant, J.L.: On the connecting thickness of arithmetical discrete planes. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 362–372. Springer, Heidelberg (2009)
9. Gérard, Y., Debled-Rennesson, I., Zimmermann, P.: An elementary digital plane recognition algorithm. *Discrete Applied Mathematics* 151, 169–183 (2005)
10. Gilbert, E.G., Johnson, D.W., Keerthi, S.S.: A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal of Robotics and Automation* 4, 193–203 (1988)
11. Jamet, D., Toutant, J.L.: Minimal arithmetic thickness connecting discrete planes. *Discrete Applied Mathematics* 157(3), 500–509 (2009)
12. Megiddo, N.: Linear programming in linear time when the dimension is fixed. *Journal of the ACM* 31(1), 114–127 (1984)
13. Reveillès, J.P.: *Géométrie discrète, calculs en nombre entiers et algorithmique*. Thèse d'état, Université Louis Pasteur, Strasbourg (1991)
14. Toussaint, G.T.: Solving geometric problems with the rotating calipers. In: *Proceedings of IEEE MELECON 1983*, Athens, Greece (1983)
15. Vapnik, V.: *Statistical Learning Theory*. Wiley, New York (1998)
16. Ziegler, G.M.: *Lectures on polytopes*. Springer, Heidelberg (1995)

Another Definition for Digital Tangents

Thierry Monteil

CNRS – Université Montpellier 2
<http://www.lirmm.fr/~monteil>

Abstract. The aim of this short note is to describe the set of finite words that appear in the cutting sequences of a smooth curve to arbitrary small scale. This language strictly contains the factors of Sturmian words, and can be decided by a linear time algorithm.

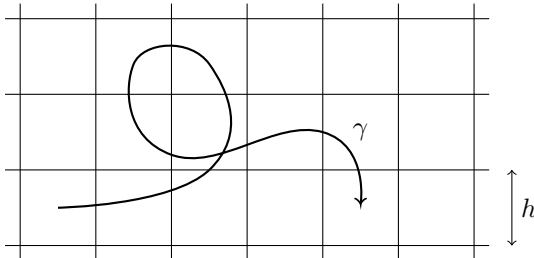
Keywords: Cutting sequence, symbolic coding, tangent estimation, multigrid convergence, digital straight segment, Sturmian word.

1 Introduction

A *smooth curve* is a map γ from a compact interval I of the real line to the plane, which is C^∞ and such that $\|\gamma'(t)\| > 0$ for any $t \in I$ (this last property is called *regularity*). Any such curve can (and will be considered to) be arc-length reparametrised (*i.e.* $\forall t \in I, \|\gamma'(t)\| = 1$).

We can approximate such a curve by drawing a square grid of width (*mesh* or resolution) h on the plane, and look at the sequence of squares that the curve meets. For a generic position of the grid, the curve γ does not hit any corner and crosses the grid transversally, hence the curve passes from a square to a square that is located either *right*, *up*, *left* or *down* of it. We record this sequence of moves and define the *cutting sequence* of the curve γ with respect to this grid as a word w on the alphabet $\{r, u, l, d\}$ which tracks the lines of the grid crossed by the curve γ .

The following picture shows a curve γ with cutting sequence *rruuldrrrd*.



Note that since the grid can be translated, a given curve may have more than one cutting sequence for a given mesh h . Our knowledge of the curve from one of its cutting sequences increases when the mesh h decreases, and when the mesh approaches 0, the local patterns of the cutting sequence allow the

digital geometers to define infinitesimal estimators (like tangents or curvature), for example by recognising maximal straight segments appearing in the cutting sequence [3] [2].

Cutting sequences associated to straight segments are known to be exactly the *balanced words* (see Section 3.5), which are also the finite factors of Sturmian words. A problem is that some non-balanced words appear at arbitrary small scale even for very regular curves. For example, the word *rruu* appears in the cutting sequence of infinitely many Bresenham circles [4] but is not balanced. This “infinitesimal noise” has the following consequence: when the mesh tends to 0, the length of the maximal segments does not necessarily tend to infinity, causing some convergence problems for the estimators.

A solution might be found in another definition of what digital tangents are: in this paper, we consider all the finite words that appear in the cutting sequences of some smooth curve for arbitrary small scale (like *rruu* for the circle). More precisely, let $F(\gamma, G)$ denote the set of factors of the cutting sequence of the curve γ with respect to the square grid G (when the curve hits a corner, the cutting sequence is not defined and we set $F(\gamma, G) = \emptyset$). We define the *asymptotic language* of γ by

$$T(\gamma) = \limsup_{mesh(G) \rightarrow 0} F(\gamma, G) = \bigcap_{\varepsilon > 0} \bigcup_{mesh(G) \leq \varepsilon} F(\gamma, G).$$

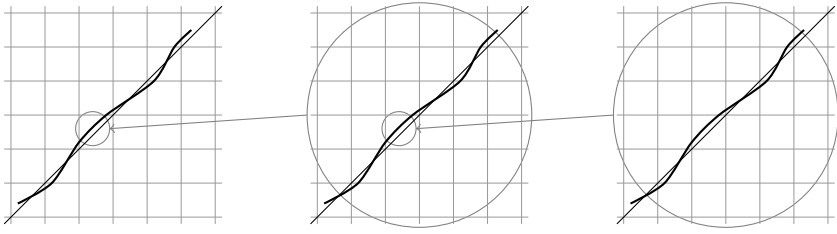
More generally, when X is a set of curves, let us denote by $T(X)$ the set $\bigcup_{\gamma \in X} T(\gamma)$. When X is the set of smooth curves, we simply denote $T(X)$ by T , and call its elements *tangent words*. The aim of this note is to characterise this language. Note that we are only interested in the language, not on the rate of convergence to it.

2 Analytic Characterisation

Proposition 1. *A word w is tangent if, and only if, for any $\varepsilon > 0$, w is the cutting sequence of a curve γ which is ε -close (for the C^1 norm) to a straight segment (the grid is fixed).*

Proof. The *only if* part is straightforward. For example, the word 00011 cannot be tangent, since, when the mesh goes to zero, it corresponds to a point where the slope of the curve is both at most 1/2 (because of the factor 000) and at least 1 (because of the factor 11).

For the *if* part, since the space C^∞ endowed with the distance defined by $d(\gamma, \delta) = \sum_{n \geq 0} 2^{\min(1, \sup_{t \in I} \|\gamma^{(n)}(t) - \delta^{(n)}(t)\|)}$ is complete, we can build a convenient curve by accumulating arbitrarily small and flat copies of a curve whose cutting sequence is w (the n th copy of the curve should be flat enough so that its first n derivatives are very small, ensuring that we get a Cauchy sequence whose limit has the desired property).



□

3 Combinatorial Description

For the sake of simplicity, let us first focus on curves going right and up, *i.e.* smooth curves such that both coordinates of $\gamma'(t)$ are positive for any t . Let us rename r and u by 0 and 1 respectively to stick to the usual notation.

3.1 Renormalisation (Desubstitution)

Balanced words are known to have a hierarchical structure, where the morphisms $\sigma_0 = (0 \mapsto 0, 1 \mapsto 10)$ and $\sigma_1 = (0 \mapsto 01, 1 \mapsto 1)$ play a crucial role [7] [5]. The same renormalisation applies to tangent words.

Proposition 2. *Let w be a finite word over the alphabet $\{0, 1\}$. The following are equivalent:*

- w is tangent.
- $\sigma_0(w)$ is tangent.
- $\sigma_1(w)$ is tangent.

Proof (sketch). It suffices to notice that applying the substitution σ_1 (resp. σ_0) to the word w geometrically corresponds to applying the linear bi-uniformly continuous bijection given by the matrix $M_0 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ (resp. $M_1 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$). Such maps preserve tangency. A clear presentation of this argument can be read in [9]. □

Hence, given a finite word w , we can “desubstitute” it by

- removing one 0 per run of 0 if 11 does not appear in w , or
- removing one 1 per run of 1 if 00 does not appear in w .

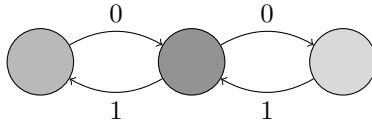
If we repeat this process as much as possible, we get a *derivated word* denoted by $d(w)$. The word w is balanced if, and only if, $d(w)$ is the empty word, and the derivation process is related to the continued fraction development of the slope of the associated straight segment.

3.2 Last Step

To finish the description of tangent words going up and right, it suffices to describe the set of derivated words that are tangent words.

Let us say that a word w is *diagonal* if one of the following equivalent conditions hold:

- w is recognised by the following automaton with three states, which are all considered as initial and accepting:

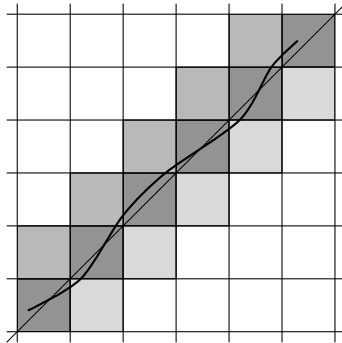


- the word w is in the language defined by the regular expression $(\varepsilon|0|1)(01|10)^*(\varepsilon|0|1)$,
- there exists an integer k such that for any prefix p of w , the difference between the number of occurrences of 0 in p and the number of occurrences of 1 in p is between k and $k + 2$, *i.e.*

$$\max\{|p|_0 - |p|_1 \mid p \text{ is a prefix of } w\} \leq \min\{|p|_0 - |p|_1 \mid p \text{ is a prefix of } w\} + 2,$$

where $|p|_i$ denotes the number of occurrences of the letter i in p .

For example, the word 0110100110 (which is not balanced) is diagonal:



Proposition 3. *A word w is tangent if, and only if $d(w)$ is diagonal.*

Proof (sketch). If $d(w)$ is not the empty word, then 00 and 11 are occurrences of $d(w)$, hence the slope of the tangent is 1 (the diagonal). The three gray diagonals correspond to the three states of the automaton defining the diagonal words. \square

3.3 Example

The word $w = 01110110110111011101$ is tangent: it can be desubstituted as 01101010110110, then 010001010, and finally $d(w) = 10011$, which is diagonal since it can be written as $(10)(01)1$.

3.4 Complexity

The *complexity* of a language L is the map that counts, for any integer n , the number of elements of L of length n . Since T contains the language defined by the regular expression $(01|10)^*$, the complexity of the language T has exponential growth, whereas the complexity of the language of balanced words has cubical growth (an explicit formula was given in [6]).

3.5 Balance

A binary word w is said to be *k-balanced* if for any two factors u and v of w of the same length, the number of 0 in u and the number of 0 in v differ by at most k . The 1-balanced words (also known as *balanced* words) correspond to the cutting sequences of discrete line segments.

We already saw that the balanced words form a strict subset of the tangent words, since 0011 is tangent but not balanced. Conversely, tangent words form a strict subset of 2-balanced words, since 00011 is 2-balanced but not tangent.

3.6 Algorithm

The combinatorial description provides a linear time algorithm that decides whether a word is tangent or not: concerning the renormalisation, we can accelerate the desubstitution procedure by removing a run equal to the length of the shortest inner run from any run of the non-isolated letter (including possible leading and trailing runs even if they have shorter length). Each such accelerated desubstitution reduces the size of the word by at least $2/3$, hence, if c denotes the complexity of the `derivate` algorithm that maps a word w to $d(w)$, we have:

$$c(n) \leq n + c((2/3)n) \leq n + (2/3)n + c((2/3)^2n) \leq \dots \leq n \sum_{k \geq 0} (2/3)^k \leq 3n .$$

The last step consist in deciding whether the obtained derived word matches the regular expression $(\varepsilon|0|1)(01|10)^*(\varepsilon|0|1)$, this can be achieved in linear time as well. A basic implementation exists in the free open source mathematical software Sage: the `is_tangent()` method is being reviewed as ticket #9877 and can be tested at <http://www.sagenb.org/home/pub/2123/>.

Moreover, any existing digital straight segment recognition algorithm based on the hierarchical structure can easily be adapted to tangent words recognition, in particular, we can construct on-line linear-time algorithms for this purpose [1]. Also, digital straight segment recognition can easily be replaced by tangent word recognition in existing digital geometry algorithms, in particular for those dealing with curve segmentation.

4 Other Classes of Curves

Let us briefly study the asymptotic language of some other classes of curves. Each class is stable by the action of invertible linear maps on the plane, hence the associated languages turn out to be stable by the renormalisation procedure described in subsection 3.1.

4.1 Continuous Curves

Proposition 4. *If X denotes the set of continuous curves (defined on a closed interval), then all the words are admissible: $T(X) = \{0, 1\}^*$ (or $T(X) = \{r, u, l, d\}^*$ if we deal with curves going in all directions).*

Proof. Given a word w , it suffices to construct a curve $\gamma : [0, 1/2] \rightarrow \mathbb{R}^2$ whose cutting sequence is w , then to glue a smaller copy of it from $[1/2, 3/4]$ to \mathbb{R}^2 , then to glue another even smaller copy from $[3/4, 7/8]$ to \mathbb{R}^2 and so on to get a curve $[0, 1] \rightarrow \mathbb{R}^2$ that admits w in its cutting sequence at arbitrary small scales. □

Note that the statistical properties of the finite words appearing in the asymptotic language of generic continuous functions have been studied (in a similar framework) in [8].

4.2 C^k Curves

Proposition 5. *If X denotes the set of C^k regular curves, then $T(X) = T$ ($1 \leq k \leq \infty$).*

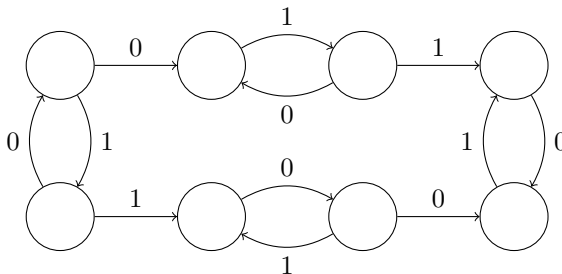
Proof. The *only if* part of the proof of Proposition 4 only uses the fact that the curve is C^1 . The C^∞ function built in the *if* part is in particular C^k for any $1 \leq k \leq \infty$. □

Hence, the asymptotic language shows some stability with respect to the regularity of the curve. This is a hint to grasp higher order notions such as curvature: we will probably have to look one step further, like the mutual organisation of those words. Note that existing methods based on maximal digital straight segments were proven to be not rigorous [10].

4.3 Analytic Curves

However, many tangent words won't appear for the more rigid class of analytic curves.

Proposition 6. *If X denotes the set of analytic regular curves, then a finite word w is in $T(X)$ if, and only if, $d(w)$ is recognised by the following automaton with eight states, which are all considered as initial and accepting:*



Proof (sketch). The tangent words oscillating at least twice should be removed. For example, the word 001100 cannot be in $T(X)$ since, if a corresponding analytic curve γ is denoted by (x, y) , then the derivative of $(x - y)$ has two close zeroes, but since the mesh can be arbitrarily small, we get an accumulation of such zeroes, which contradicts the analyticity of the map $(x - y)'$. \square

Note that those remaining words can all be found in $T(\mathcal{C})$, where \mathcal{C} denotes the set of planar circles (winding both clockwise and counterclockwise).

4.4 Smooth Curves Defined on an Open Interval

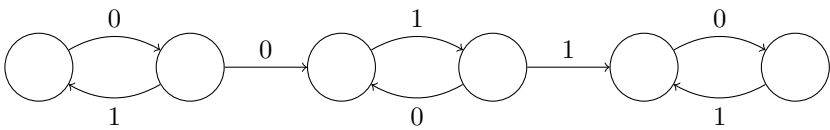
To get a uniform control on the tangents of the curve, we assumed the compactness of the interval on which the smooth curve is defined. Here is why this assumption was necessary.

Proposition 7. *If X denotes the set of regular smooth curves defined on an open interval, then $T(X) = \{0, 1\}^*$ (or $T(X) = \{r, u, l, d\}^*$ if we deal with curves going in all directions).*

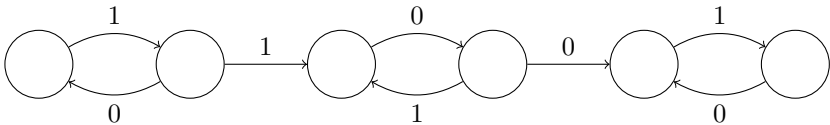
Proof. As in the proof of Proposition 4, we can accumulate any noise near a boundary of the interval, since there is no need to ensure derivability at the endpoint. \square

4.5 Smooth Curves with Nowhere Zero Curvature

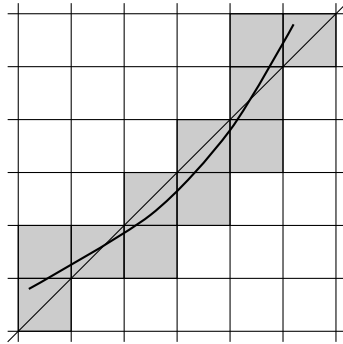
Proposition 8. *If X (resp. Y) denotes the set of smooth curves whose curvature is positive (resp. negative), then a finite word w is in $T(X)$ (resp. $T(Y)$) if, and only if, $d(w)$ is recognised by the following automaton with six states, which are all considered as initial and accepting:*



and respectively for smooth curves with negative curvature:



For example, the word 1001010110 appears in the cutting sequence of a smooth curve with positive curvature:



We can notice that the asymptotic language of analytic regular curves is the union of those two languages, which is the asymptotic language of smooth curves with nowhere zero curvature, showing again some stability with respect to the considered class of curves.

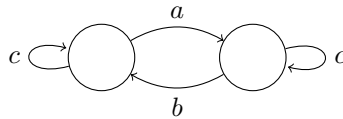
5 Curves Going in All Directions

Let us finish by the complete characterisation of tangent words without any assumption on the direction of the curve.

The four letters $\{r, u, d, l\}$ cannot all appear simultaneously in a tangent word.

If only two letters appear in a tangent word and are consecutive for the cyclic order $r < u < l < d$, then, if we replace them by 0 and 1, we have the same characterisation as above.

If two non-consecutive letters a and b appear in a tangent word w , we are in the case where the tangent is vertical or horizontal, and this case is similar to the last step described before (no renormalisation must be done): there exists a letter c , different from a and b , such that w is recognised by the following automaton with two states, both initial and accepting:



That is, after removing the occurrences of c in w , the letters a and b are alternating. Equivalently the word w is in the language defined by the regular expression $c^*(\varepsilon|b)(c^*ac^*bc^*)^*(\varepsilon|a)c^*$.

6 Conclusion

We introduced and described the asymptotic language of smooth curves, *i.e.* the set of words that can survive in the cutting sequence of a smooth curve when the grid mesh goes to zero, and discussed some of its properties. We saw that those words are closely related to the tangents of the curve.

So, we can use them to change one of the most basic primitive for discrete geometry, that of discrete tangency: instead of using digital straight segment, we could use tangent words. One possible advantage is that the smallest length of maximal tangent words one can find in a smooth curve goes to infinity when the grid mesh goes to zero, whereas it may stay bounded for straight segments.

What can be done with this?

In terms of complexity, most tangent words are not balanced. However, we can notice a kind of prevalence of balanced words among tangent words. For example, the intersection of the asymptotic languages of all closed smooth regular curves is the set of balanced words (those are the most “stable”). Hence, the study of probabilistic aspects of the occurrences of tangent words should be interesting.

References

1. Creutzburg, E., Hübler, A., Wedler, V.: On-line erkennung digitaler geradensegmente in linearer zeit. In: Proceedings of GEO-BILD 1982, Wiss. Beitrage der FSU Jena, pp. 48–65 (1982)
2. Hermann, S., Klette, R.: A comparative study on 2d curvature estimators. In: International Conference on Computing: Theory and Applications, pp. 584–589 (2007)
3. Klette, R., Rosenfeld, A.: Digital straightness—a review. *Discrete Appl. Math.* 139(1-3), 197–230 (2004), <http://dx.doi.org/10.1016/j.dam.2002.12.001>
4. Kulpa, Z.: On the properties of discrete circles, rings, and disks. *Computer Graphics and Image Processing* 10(4), 348–365 (1979), <http://www.sciencedirect.com/science/article/B7GXF-4K9JH6S-4/2/93cd964fb7568d8161432f0401fd4159>
5. Lothaire, M.: Algebraic combinatorics on words, *Encyclopedia of Mathematics and its Applications*. In: Berstel, J., Séébold, P. (eds.) *Sturmian Words*, vol. 90. Cambridge University Press, Cambridge (2002)
6. Mignosi, F.: On the number of factors of Sturmian words. *Theoret. Comput. Sci. (Algorithms Automat. Complexity Games)* 82(1), 71–84 (1991), <http://dx.doi.org/10.1016/0304-39759190172-X>
7. Fogg, N.P.: Substitutions in dynamics, arithmetics and combinatorics. *Lecture Notes in Mathematics*, vol. 1794, p. 402. Springer, Berlin (2002), <http://dx.doi.org/10.1007/b13861>, chapter 6, *Sturmian Sequences* (by Pierre Arnoux)
8. Rojas, C., Troubetzkoy, S.: Coding Discretizations of Continuous Functions (2009), preprint, available at <http://arxiv.org/abs/0907.2299>
9. Smillie, J., Ulcigrai, C.: Symbolic coding for linear trajectories in the regular octagon (2009), preprint, available at <http://arxiv.org/abs/0905.0871>
10. de Vieilleville, F., Lachaud, J.-O., Feschet, F.: Convex digital polygons, maximal digital straight segments and convergence of discrete geometric estimators. *J. Math. Imaging Vision* 27(2), 139–156 (2007), <http://dx.doi.org/10.1007/s10851-007-0779-x>

Associating Cell Complexes to Four Dimensional Digital Objects

Ana Pacheco and Pedro Real

Dpto. Matematica Aplicada I, E.T.S.I. Informatica,
Universidad de Sevilla,
Avda. Reina Mercedes, s/n 41012 Sevilla (Spain)
<http://ma1.eii.us.es/>

Abstract. The goal of this paper is to construct an algebraic-topological representation of a 80-adjacent doxel-based 4-dimensional digital object. Such that representation consists on associating a cell complex homologically equivalent to the digital object. To determine the pieces of this cell complex, algorithms based on weighted complete graphs and integral operators are shown. We work with integer coefficients, in order to compute the integer homology of the digital object.

Keywords: digital object, integer homology, integral operator, weighted complete graph.

1 Introduction

Several techniques as magnetic resonance (MR) images and computed tomography (CT) images allow to represent voxel-based digital objects. The homology can be used to obtain topological information of such these objects, but the homological study cannot be made directly over the digital objects. In this sense, it is necessary to apply a thresholding process, in such way, the representation of the voxel-based digital object is made using two colors black and white, which represent (respectively) the elements of the object and the part of the space where such object is not included. The homological study does not only consist on computing Betti numbers, but on determining connected components, “holes”, tunnels and cycles (closed curves) in the object. To be extendible previous techniques to higher dimensions is an important goal in computer vision.

Integer homology information of a subdivided 4D object (consisting on a set of contractile “bricks” which are glued in a “coherent” manner) is given in this paper by a boundary operator and a “homology” operator for any finite linear combination (with integer coefficients) of bricks, such that, in particular, the boundary of the boundary (resp. the “homology” of the “homology”) is zero. Both of them operators can be expressed in terms of arrows acting over the cells of each “brick”. For example, the boundary operator (with integer coefficients) of a triangle is an alternate sum of its edges; and the “homology” operator of this triangle is a linear map describing in an algebraic way the contractibility of the triangle to one of its vertices (see Figure 1 for more details).

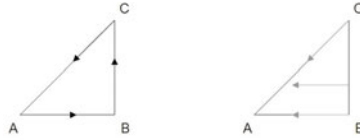


Fig. 1. Given a triangle of vertices A, B, C , boundary and “homology” operators can be expressed with arrows defined over its cells: (a) representation in terms of arrows of the boundary operator given by the formula $\partial(ABC) = \partial_0(ABC) - \partial_1(ABC) + \partial_2(ABC) = BC - AC + AB$; and (b) representation in terms of arrows of the “homology” operator (which expresses the contractibility of the triangle to one point) as composition of the integral operators $\phi_1(B) = AB, \phi_2(C) = AC, \phi_3(BC) = ABC$

In this paper, we want to extend the techniques shown in [4-7, 10, 11] to dimension 4. In order to get our goal, we show a method to construct a cell complex homologically equivalent to a 4-dimensional digital object. This cell complex is built piece by piece. The bricks which compose the cell complex are obtained (up to isometry) determining firstly their vertices, and then computing the convex hull of such set of vertices by deforming the unit hypercube with integral operators. In this way, the boundary and contractibility operators of each one of the bricks are inherited of the respective boundary and contractibility operators of the unit hypercube. In Figure 2 we can see the digital object, the associated cell complex and the extracted homological information.

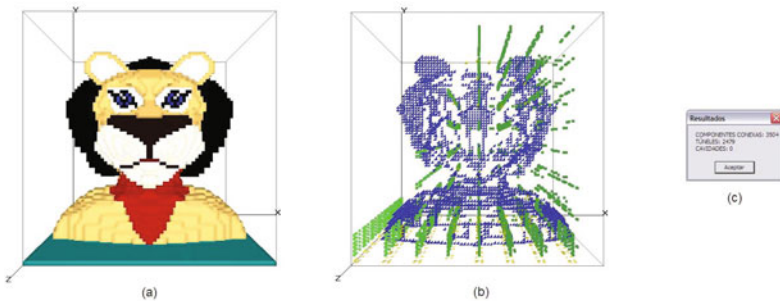


Fig. 2. (a) Digital object; (b) Cell complex associated to the digital object, 0-cells, 1-cells, 2-cells and 3-cells are shown in green, brown, yellow and blue respectively; (c) Homological information expressed in terms of connected components (3504), tunnels (2479) and cavities (0)

The shown method, presents some computational improvements respect to similar methods such as the shown in [8]. Here, unnecessary data corresponding to the simplicialization of the convex hull of the unit hypercube are not to saved, and consequently the convex hull of each one of the bricks is obtained directly. Moreover, the number of integral operator to compute the convex hull of each

brick decreases, since the number of cells of the unit hypercube is smaller than the number of simplices of the simplicialized unit hypercube.

The structure of the paper is the following: in Section 1, we introduce concepts which appear along the paper; in Section 2, we explain our framework and we develop algorithms to associate the cell complex to a given digital object; and finally in Section 3, we show both as several examples of the 402 configurations of points obtained in dimension 4 (applying the algorithms of the previous section) as the way of transferring the boundary and “homology” operators.

2 Preliminaries

As we have commented in Introduction, we show a method to construct (piece by piece) dimensional cell complexes associated to 4–dimensional digital objects using as tools graphs and integral operators. So that, it is necessary to introduce several concepts such these digital objects, cell complexes, isomorphic graphs, integral operators...

The term *digital object* is used for denoting an identifiable item of structured information in digital form within a network-based computer environment. A digital object is a set of sequences of bits or elements, each one of these constitutes structured data interpretable by a computational facility, at least one of the sequences denoting a unique, persistent identifier for that object.

In this paper, we associated to each digital object a mathematical object very used in topology, called cell complex.

A *cell complex* is a set $K = \{K_{(q)}\}_{q \geq 0}$ of cells satisfying two conditions: (1) every face of a cell is a cell; and (2) if σ and σ' are cells, then their intersection is a common face of both (or empty). A cell complex is denoted by (K, ∂) where K is the set of cells and ∂ is a map indicating how to join the cells and satisfies $\partial\partial = 0$. The boundary operator shows a relation between cells of different dimensions in order to capture the topology of the cell complex. For example, the boundary of a 1–cell c consists of its two end-points, so using binary coefficients $\partial(c) = A + B$; using integer coefficients, the direction of the edge matters ($AB \neq BA$), so we define $\partial(c) = B - A$. In dimension 2, given a cell complex K whose 0–cells are A, B ; 1–cells are $a = AB, b = CB, c = AC$; and the 2–cell is $\tau = ABC$; the boundary is defined as a linear combination of its faces $\partial\tau = AB + BC + CA = a + b - c$.

A graph can be seen as a cell complex of dimension 1, that is, a set of vertices and edges and the relations between them. Graphs are used in Algorithm 1 to obtain the 0–cells of the cell complex associated to a given digital object.

A special type of graphs is the family of *complete graphs*. A graph is complete if every pair of distinct vertices is connected by an edge. Moreover, if we associate to each edge a weight, we obtain the family of *weighted complete graphs* (a graph of this family is associated to each subset of vertices of the unit hypercube in order to determine the non-isometric configurations). See Figure 3 (a) for an example of weighted complete graph.

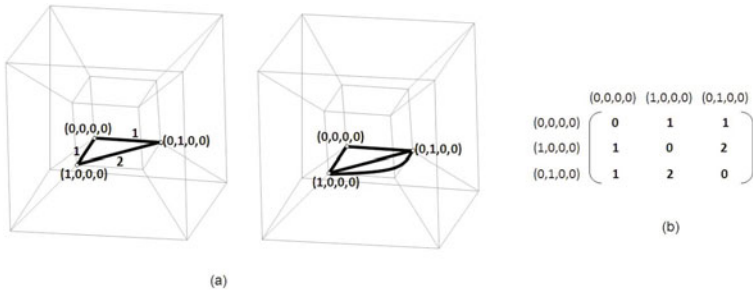


Fig. 3. (a) Two representations of the same situation in terms of graphs (using complete graphs with multiple edges or weighted complete graphs); and its (b) Adjacency matrix

In a natural way, we say G_1 and G_2 are *isomorphic graphs* if there exists a bijection between the set of its vertices which preserves adjacencies (if two vertices of G_1 are joined by an edge, then their images by the bijection must be two vertices of G_2 joined by an edge).

An usual tool to represent adjacency relations between vertices of a graph is the *adjacency matrix*. The adjacency matrix of a graph with n vertices is an $n \times n$ matrix $A = (a_{i,j})$ in which the entry $a_{i,j} = m$ if there are m edges from vertex i to vertex j , and it is 0 if there is no edge from vertex i to vertex j . Figure 3 shows an example of a graph together to its adjacency matrix.

A stronger concept than isomorphism is isometry. An *isometry* is a distance-preserving map between spaces. For example, the isometries in a 3-dimensional space are rotations, translations and symmetries.

Information about graph theory can be founded in [1].

Other tools used for obtaining the cell complex associated to a digital object are the integral operators (see [3]).

Given a cell complex (K, ∂) and two cells $a \in K_q$ and $b \in K_{q+1}$ (a is q -face of b), an *integral operator* $\phi_{a,b} : (K_q) \rightarrow (K_{q+1})$ is a linear map satisfying: (1) $\phi_{a,b}(a) = b$ and $\phi_{a,b}(c) = 0$ for $c \in K$ different from a ; and (2) $\phi_{a,b}\partial\phi_{a,b} = \phi_{a,b}$.

Roughly speaking, an integral operator can be seen as an elementary algebraic thinning operation, allowing the deformation of a cell complex to smaller one (eliminating the cells a and b) with isomorphic homologies. The integral operator $\phi_{a,b}$ can be represented as an arrow from the lower dimension cell a until the higher dimension cell b . In this sense, we say that a cell complex is *contractible* if it has the same topology of a point. For example, the unit hypercube is contractible and its contractibility can be measured in algebraic terms (specifying that the unit hypercube and a point of this one have isomorphic homology groups) by the sequence of integral operators shown in Figure 4.

A *chain contraction* $(f, g, \phi) : (K, \partial) \Rightarrow (K', \partial')$ between two cell complexes is a set of three morphisms $f : \mathcal{C}(K_q) \rightarrow \mathcal{C}(K'_q)$ (projection), $g : \mathcal{C}(K'_q) \rightarrow \mathcal{C}(K_q)$

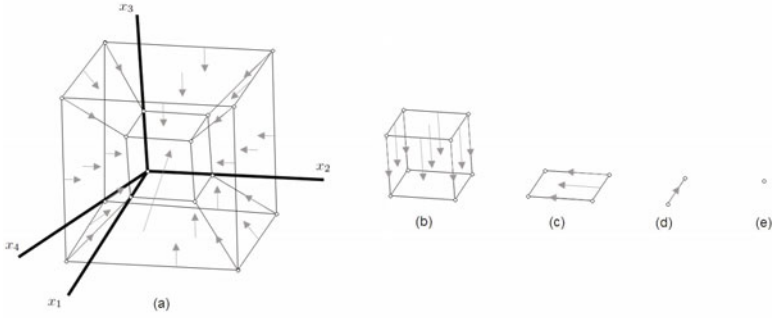


Fig. 4. In order to contract the unit hypercube to the point $(0, 0, 0, 0)$, we define a set of integral operators which are represented by arrows whose: (a) direction coincides with the result of intersecting the hypercube with the hyperplane $x_4 = 0$; (b) direction coincides with the result of intersecting $x_4 = 0$ with $x_4 = x_3 = 0$; (c) direction coincides with the result of intersecting the plane $x_4 = x_3 = 0$ with the line $x_4 = x_3 = x_2 = 0$; and (d) direction coincides with the result of intersecting the line $x_4 = x_3 = x_2 = 0$ with the point $x_4 = x_3 = x_2 = x_1 = 0$. This contraction can be seen as the projection on the coordinate axes

(inclusion) and $\phi : \mathcal{C}(K_q) \rightarrow \mathcal{C}(K_{q+1})$ (homotopy operator), where $\mathcal{C}(K_q)$ (resp. $\mathcal{C}(K'_q)$) denotes the set of cell of dimension q of the cell complex (K, ∂) (resp. (K', ∂')) satisfying the following conditions: (a) $\pi = gf = id_C - \partial\phi - \phi\partial$; (b) $fg = id_{C'}$; (c) $f\phi = 0$; (d) $\phi g = 0$; (e) $\phi\phi = 0$.

In Figure 5 we can see an example about integral operators and chain contraction.

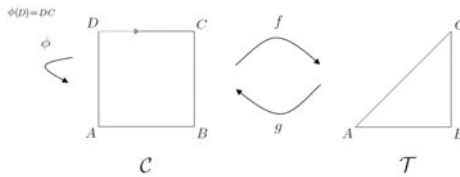


Fig. 5. Integral operators deforming a square \mathcal{C} to a triangle \mathcal{T} . The boundary operator of the triangle is computed starting from the boundary operator of the square using the maps which compose the chain contraction as follows $\partial'(\mathcal{T}) = f\partial g(\mathcal{T}) = f\partial(\mathcal{C}) = f\partial(ABCD) = f(AB + BC + CD + DA) = f(AB + BC - DC - AD) = (id - \partial\phi - \phi\partial)(AB + BC - DC - AD) = (AB + BC - DC - AD) - 0 - \phi(B - A + C - B - C + D - D + A) = AB + BC - DC - AD = AB + BC - (AD + DC)$.

As we have commented in Introduction, the boundary (resp. “homology”) operator of the hypercube determines the boundary (resp. “homology”) operator of the pieces of the cell complex homologically equivalent to the given digital object. Lemma 1 shows the formula to compute the boundary of a hypercube. See Figure 6 for more details.

Lemma 1. *Let $H = L_1 \times L_2 \times L_3 \times L_4$ be a hypercube written as the cartesian product of four unit segments. The boundary operator of H is given by (1):*

$$\partial_H = \partial L_1 \times L_2 \times L_3 \times L_4 - L_1 \times \partial L_2 \times L_3 \times L_4 + L_1 \times L_2 \times \partial L_3 \times L_4 - L_1 \times L_2 \times L_3 \times \partial L_4$$

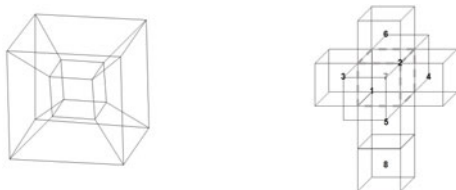


Fig. 6. On left, the tesseract representing a hypercube. On right, the spatial development of a hypercube; the cubes labeled by an odd (resp. even) number have positive (resp. negative) sign in formula (1).

3 Constructing the Cell Complex Associated to a Given 4–Dimensional Digital Object

In this section, the grid and the neighborhood between the points of this one are fixed; and the algorithms to obtain the cell complex associated to a 4–dimensional digital object are developed.

3.1 Establishing the Grid and the Neighboring between the Points

In order to work with digital objects it is necessary to fix a grid as well as the relations between the points of the grid. Our grid is divided into hypercubes (whose intersection is a 3–dimensional cube of 8 mutually 26–adjacent 4–dimensional points) formed by 16 mutually 80–adjacent 4–dimensional points (we work with possible maximal adjacency between points). An example of this division is shown in Figure 7

This grid is a natural extension to dimension 4 of the grid used in [9], where similar techniques were developed in order to associate cell complexes to 3–dimensional digital objects.

Once established the grid, the initial digital object is embedded in it, so a subdivision into hypercubes of the object is obtained. Applying a thresholding process, we have a digital object subdivided into hypercubes such way that the vertices of each hypercube which are (resp. are not) points of the object are black (resp. white).

Now, the idea is to work each one of these hypercubes with black and white points separately, in order to obtain the pieces of the cell complex associated to the initial object. After, we must join each one of the obtained pieces to compose the cell complex homologically equivalent to the given digital object.

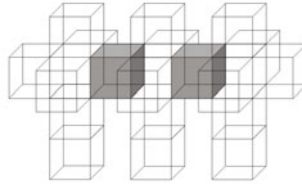


Fig. 7. Spatial development of a sequence of hypercubes formed by 16 mutually 80-adjacent 4-dimensional points, where the intersection between the pairs is a cube of 8 mutually 80-adjacent 4-dimensional points

3.2 Cell Complex Obtention

The main goal of this subsection is the construction of the cell complex associated to a given 4-dimensional digital object. The bricks which compose the cell complex are obtained (up to isometry) determining firstly their vertices (Algorithm 1), and then computing the hull of such set of vertices by deforming the unit hypercube with integral operators (Algorithm 2). In this way, the boundary and “homology” operators of each one of the bricks are inherited of the respective boundary and “homology” operators of the unit hypercube.

In order to determine the vertices of the pieces which compose the complex, we develop an algorithm based on isometric graphs (1-dimensional cell complexes) which allows to compute the non-isometric configurations of c points of the unit hypercube, for $c = 0, \dots, 16$. This algorithm associates to each set of points of the unit hypercube a weighted complete graph whose vertices are the points of the set and whose edges are determined by the number of different coordinates between each pair of points. For example, the graph represented in Figure 3 is associated to the set of vertices $\{(0, 0, 0, 0), (1, 0, 0, 0), (0, 1, 0, 0)\}$.

Taking into account previous association is natural to establish Definition 1.

Definition 1. *Two subsets of vertices of the unit hypercube are isometric if and only their respective associated graphs are isometric.*

In order to see the consistency of Definition 1 we must prove Theorem 1.

Theorem 1. *If two subsets of vertices S_1 and S_2 of the unit hypercube are isometric, then there exists a linear isometry $f : \mathbb{R}^4 \rightarrow \mathbb{R}^4$ which sends S_1 into S_2 .*

Proof. Taking into account complements (the number of subsets of c vertices of the unit hypercube is the same that the number of subsets of $16 - c$ vertices, with $0 \leq c \leq 16$), it is only necessary to prove the result for subsets with at less 8 vertices of the unit hypercube.

The idea of the proof is the following:

- If $c > 8$, then with the vertices $\{v_0, \dots, v_{c-1}\}$ we can construct a basis of \mathbb{R}^4 composed by the vectors $\{v_i - v_0\}_{1 \leq i \leq c-1}$, and the problem would be solved by linearity.

- If $c = 8$ and we can construct a basis of \mathbb{R}^4 with the vectors $\{v_i - v_0\}_{1 \leq i \leq 7}$, then the problem would be solved by linearity.

- If $c = 8$ and we cannot construct a basis of \mathbb{R}^4 with the vectors $\{v_i - v_0\}_{1 \leq i \leq 7}$, then the vertices $\{v_0, \dots, v_7\}$ are in a cubic face of the hypercube and all the cubes are isometric.

Using previous ideas, we develop Algorithm 1 which compares the graphs built starting from subsets of points of the unit hypercube, saving only non-isometric graphs. So, identifying non-isometric subsets of points with their respective associated graphs, we obtain all the vertices of the non-isometric bricks which can compose the cell complex associated to a 4-dimensional object. By complement configurations, it is only necessary to use the algorithm for subsets with at less 8 points.

Algorithm 1

```

Input: set  $(V_H)$  of the 16 vertices of the unit hypercube.
//  $\Omega$ : empty list to save vertices of non-isometric graphs.
Output: non-isometric configurations of vertices of the unit hypercube.
begin
  for  $c=8, \dots, 16$  do
    Construct an ordered set  $\Omega_c$  with all the subsets of  $c$  vertices of  $V_H$ .
    for  $\omega \in \Omega_c$  do
       $G_\omega$  weighted complete graph with adjacency matrix
       $M_\omega = ((m_\omega)_{ij})$  where  $(m_\omega)_{ij} = k_{ij}$ ,
       $k_{ij}$  is the number of different coordinates between  $v_i, v_j \in V_H$ 
      while  $\omega \in \Omega_c \ \& \ \omega' \in \Omega_c \ \& \ \omega' < \omega$  do
        if  $G_\omega$  and  $G_{\omega'}$  are isometric then
           $\omega$  and  $\omega'$  are isometric
           $\Omega_c = \Omega_c - \{\omega\}$ 
        end if
        if  $G_\omega$  and  $G_{\omega'}$  are not isometric then
          Update  $\omega'$ 
        end if
      end while
    end for
     $\Omega = \Omega \cup \Omega_c$ 
  end for
end

```

Note 1. Algorithm 1 is a way to compute marching cube configuration in 4D.

Once obtained the vertices of the non-isometric bricks which can compose the searched cell complex, the idea is to use integral operators (in a right manner) to deform the unit hypercube in the convex hull of each one of the configurations of points.

Before to show the algorithmic process to define the set of integral operators to deform the unit hypercube, we need to establish a direction over the arrows which represent these integral operators. Indeed, the choice of the direction of

these arrows is arbitrary. We have decided to chose the set of these integral operators as a subset of the sequence (ordered set) of integral operators given in Proposition [11](#)

Proposition 1. *The sequence of integral operators ϕ_H , which computes the combinatorial contractibility of the unit hypercube to the point $(0,0,0,0)$ is represented by arrows whose directions coincide with the result of:*

- *Intersecting the hypercube with the hyperplane of dimension 3, $x_4 = 0$ (let us note that the result of the intersection is the hyperplane $x_4 = 0$).*
- *Intersecting the hyperplane $x_4 = 0$, with the plane composed by the equations $x_4 = x_3 = 0$.*
- *Intersecting the plane $x_4 = x_3 = 0$, with the line composed by the equations $x_4 = x_3 = x_2 = 0$.*
- *Intersecting the line $x_4 = x_3 = x_2 = 0$ with the point $x_4 = x_3 = x_2 = x_1 = 0$.*

In Figure [4](#) can be seen a constructive proof of the deformation of the hypercube to the point $(0,0,0,0)$ by ϕ_H .

Once determined the direction of the arrows which represent all integral operators which we can chose to deform each hypercube associated to a configuration of points ω ; we show an algorithm which allows us to decide which of these integral operators we must chose to obtain the different pieces which can compose the cell complex associated to the initial object (see Algorithm 2).

Algorithm 2

Input: A configuration of points ω (obtained using Algorithm 1).

Hypercube H_ω associate to a ω configuration.

Sequence of integral operators ϕ_H .

Output: Hull of the points of ω .

begin

for every 0–cell $\sigma \in H$ **do**

if $\sigma \notin \omega$ **then**

$\phi(\sigma) = \phi_H(\sigma)$

end if

end for

for every degenerate cell δ whose 0–cells are in ω **do**

$\phi(\delta) = \phi_H(\delta)$

end for

return $\phi(\omega)$

end

Note: We consider degenerate d –cells as those with at most $d - 1$ –faces.

Algorithm 2 is divided into two stages: (1) to apply ϕ_H (which computed the contractibility of the unit hypercube to the point $(0,0,0,0)$) to the white points of each hypercube associated to a configuration ω ; (2) to eliminate degenerate cells (if they are appeared in previous stage) applying on them ϕ_H .

Note 2. Let us observe that Algorithm 2 determines the hull (non convex hull) of the points. In order to obtain a convex polytope, we must attach an edge for each non-plane face. This process can be made implementing a simple algorithm of recognition of non-plane faces and attaching (in each one of them) an edge whose vertices are the neighbors of the point $p_i \in \omega$ of the non-plane face. Consequently, each non-plane face is transformed into two plane faces.

The pieces which compose the cell complex associated to a given 4–dimensional digital object have been determined (up to isometry). Now, we only need to join these pieces to obtain the searched cell complex. Let us note that such that complex is homologically equivalent to the given 4–dimensional digital object according to Proposition 1 in [3].

4 Conclusions and Results

In this paper, we have shown the process to obtain a cell representation of a 4–dimensional digital object where the topological information is saved. We have obtained 402 configurations (running Algorithm 1 in the symbolic computation package Mathematica) which are the pieces that can compose the cell complex associated to a 4–dimensional digital object. In Figure 8 we can see several examples of the pieces with 14 vertices.

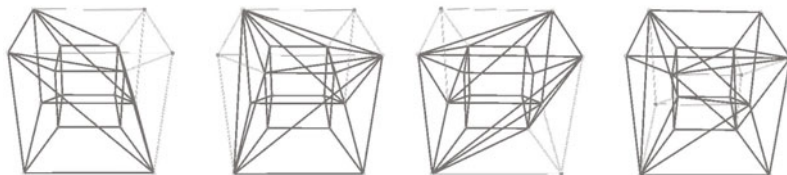


Fig. 8. Convex hulls of the 4 non-isometric bricks of 14 vertices of the unit hypercube

Below, we show with detail each one of the stages of the described process in previous section with one of the 19 non-isometric configurations of 4 vertices of the unit hypercube, and we also show the transference of the boundary (resp. “homology”) operator of the unit hypercube to the single one non-isometric configuration of 15 vertices of the unit hypercube.

4.1 Results Obtained for Configurations of 4 Vertices of the Unit Hypercube

Firstly, using Algorithm 1 with $c = 12$ (whose complementary configurations corresponding to the subsets of 4 vertices of the unit hypercube) we obtain 19 non-isometric bricks of 4 vertices.

Now, we must determine the hull of the 19 non-isometric subsets of vertices using Algorithm 2. Particularly, applying Algorithm 2 to the configuration of points $\{\{0, 0, 0, 0\}, \{1, 0, 0, 0\}, \{0, 1, 0, 0\}, \{1, 1, 1, 0\}\}$ (see Figure 9 (a)), we obtain a sequence of integral operators which deforms the unit hypercube in Figure 9 (b).

Let us observe that the obtained cell complex applying Algorithm 2, is not a tetrahedron (convex hull of the subset of points), so we must attach the edge $\langle (0, 0, 0, 0), (1, 1, 1, 0) \rangle$ as consequence of transforming the non-plane face $\langle (0, 0, 0, 0), (1, 0, 0, 0), (0, 1, 0, 0), (1, 1, 1, 0) \rangle$ in two plane faces $\langle (0, 0, 0, 0), (1, 0, 0, 0), (1, 1, 1, 0) \rangle$ and $\langle (0, 0, 0, 0), (0, 1, 0, 0), (1, 1, 1, 0) \rangle$ (see Figure 9(c)).

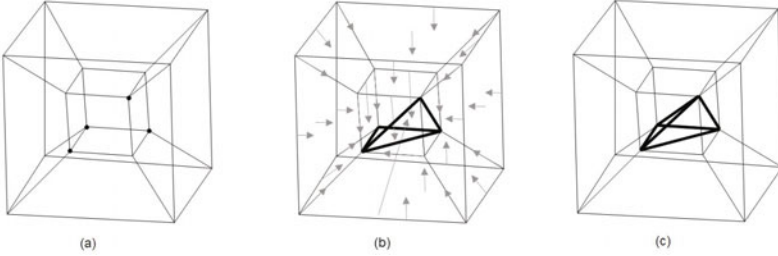


Fig. 9. (a) Vertices of one of the 19 non-isometric configurations of 4 vertices of the unit hypercube; (b) List of integral operators which deform the unit hypercube on the hull of the points; (c) Attaching the edge $\langle (0, 0, 0, 0), (1, 1, 1, 0) \rangle$ corresponding to the non-plane face $\langle (0, 0, 0, 0), (1, 0, 0, 0), (0, 1, 0, 0), (1, 1, 1, 0) \rangle$, the convex polytope is obtained.

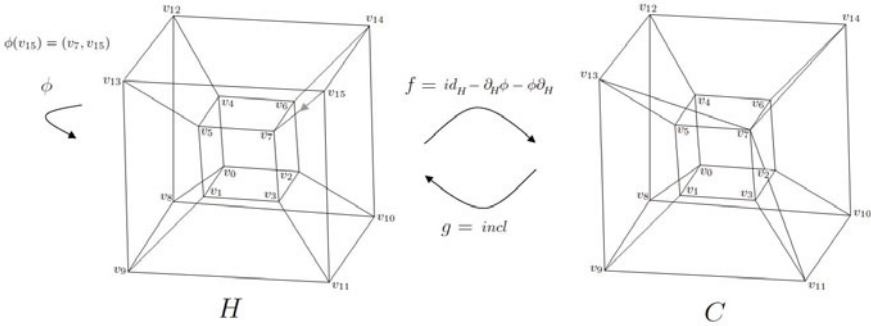


Fig. 10. The boundary operator of C is given by $\bar{\partial}(C) = f\partial g(C) = f\partial(H) = (1 - \partial\phi - \phi\partial)(\partial(H)) = -(v_7, v_5, v_4, v_{13}, v_{12}, v_7, v_{15}, v_1) + (v_{11}, v_{10}, v_6, v_{14}, v_8, v_3, v_2, v_0) + (v_{11}, v_6, v_5, v_{14}, v_{13}, v_7, v_{15}, v_3) - (v_{10}, v_9, v_4, v_{12}, v_8, v_2, v_1, v_0) + (v_{10}, v_6, v_4, v_{14}, v_{12}, v_7, v_{15}, v_2) - (v_{11}, v_9, v_5, v_{13}, v_8, v_3, v_1, v_0) - (v_6, v_5, v_4, v_7, v_3, v_2, v_1, v_0) + (v_{11}, v_{10}, v_9, v_{14}, v_{13}, v_{12}, v_{15}, v_8)$, where ϕ operator is obtained using Algorithm 2 and $\partial(H)$ is determined in Lemma 1. In an analogous way, the “homology” operator of C is given by the formula $\bar{\phi}(C) = f\phi_H g(C) = f\phi_H(H) = (1 - \partial\phi - \phi\partial)(\phi_H(H))$, where ϕ operator is obtained using Algorithm 2 and ϕ_H is determined in Proposition 1

4.2 Results Obtained for Configurations of 15 Vertices of the Unit Hypercube

In Figure 10, we show a detailed example where the boundary (resp. “homology”) operator of the single non-isometric configuration of 15 points, denoted by C ,

is computed starting from the boundary (resp. “homology”) operator of the unit hypercube, denoted by H , determined in Lemma 1 (resp. Proposition 1) and the maps of the chain contraction which relates H and C . The boundary (resp. “homology”) operator of the other configurations is computed in the same way, using the corresponding maps of the chain contraction which relates the configuration with the unit hypercube.

References

1. Diestel, R.: Graph Theory. Springer, Heidelberg (2005)
2. Fristch, R., Piccinini, R.A.: Cellular structure in topology. Cambridge University Press, Cambridge (1990)
3. Gonzalez-Diaz, R., Jimenez, M.J., Medrano, B., Molina-Abril, H., Real, P.: Integral Operators for Computing Homology Generators at Any Dimension. In: Ruiz-Shulcloper, J., Kropatsch, W.G. (eds.) CIARP 2008. LNCS, vol. 5197, pp. 356–363. Springer, Heidelberg (2008)
4. Herman, G.T.: Discrete multidimensional Jordan surfaces. CVGIP: Graphical Models and Image Processing 54, 507–515 (1992)
5. Kenmochi, Y., Imiya, A., Ichikawa, A.: Boundary Extraction of Discrete Objects. Computer Vision and Image Understanding 71, 281–293 (1998)
6. Kalvin, A.D., Cutting, C.B., Haddad, B., Noz, M.E.: Constructing topologically connected surfaces for the comprehensive analysis of 3D medical structures. In: Proc. of SPIE, vol. 1445, pp. 247–258 (1991)
7. Lorensen, W.E., Cline, H.E.: Marching cubes: A high-resolution 3D surface construction algorithm. Computer Graphics 21, 163–169 (1988)
8. Pacheco, A., Real, P.: Getting topological information for a 80-adjacency doxel-based 4D volume through a polytopal cell complex. In: Bayro-Corrochano, E., Eklundh, J.-O. (eds.) CIARP 2009. LNCS, vol. 5856, pp. 279–286. Springer, Heidelberg (2009)
9. Pacheco, A., Real, P.: Polyhedrization, homology and orientation. Progress in Combinatorial Image Analysis, 151–164 (2009)
10. Udupa, J.K.: Multidimensional digital boundaries. CVGIP: Graphical Models and Image Processing 56, 311–323 (1994)
11. Voss, K.: Discrete Images, Objects, and Functions in Z^n . Algorithms and Combinatorics (1987)

Metric Bases for Polyhedral Gauges

Fabien Rebatel and Édouard Thiel

Laboratoire d'Informatique Fondamentale de Marseille (LIF, UMR 6166),
Aix-Marseille Université,
163 Avenue de Luminy, Case 901, 13288 Marseille cedex 9, France
{Fabien.Rebatel,Edouard.Thiel}@lif.univ-mrs.fr

Abstract. Let (W, d) be a metric space. A subset $S \subseteq W$ is a resolving set for W if $d(x, p) = d(y, p)$ for all $p \in S$ implies $x = y$. A metric basis is a resolving set of minimal cardinality, named the metric dimension (of W). Metric bases and dimensions have been extensively studied for graphs with the intrinsic distance, as well as in the digital plane with the city-block and chessboard distances. We investigate these concepts for polyhedral gauges, which generalize in the Euclidean space the chamfer norms in the digital space.

Keywords: metric basis, metric dimension, resolving set, polyhedral gauge, chamfer norms, discrete distance, distance geometry.

1 Introduction

Distance geometry is the characterization and study of sets of points based on the distance values between member pairs. A general program is laid out in [2]. We investigate in this paper the classical notions of metric dimension and metric basis in the context of digital geometry, for the class of polyhedral distance gauges which generalize the chamfer (or weighted) norms.

Let W be a set endowed with a metric d , and take $S = (p_1, p_2, \dots, p_k)$ an ordered subset of elements in W . These elements are usually called points, vertices, anchors or landmarks in the literature. The *representation* of $q \in W$ with respect to S is the k -tuple $r(q|S) = \{d(p_1, q), d(p_2, q), \dots, d(p_k, q)\}$, also called the S -coordinates of q . The set S is a *resolving set* (or locating set) for W if every two elements of W have distinct representation. The *metric dimension* (or simply dimension, or location number) $\dim(W)$ is the minimum cardinality of a resolving set for W . A resolving set having minimal cardinality is called a *metric basis* (or reference set) for W .

The metric dimension has been extensively studied in graph theory. Using the intrinsic metric, Harary and Melter gave in [9] the metric dimension of any path, complete graph, cycle, wheel and complete bipartite graph, and proposed an algorithm for finding a metric basis of a tree T , which gives an explicit formula for the dimension of T .

Khuller *et al.* showed in [13] that the intrinsic metric dimension of trees can be efficiently solved in linear time. All graphs having dimension 1 are paths.

A graph having dimension 2 cannot have the complete 5-clique K_5 nor the complete bipartite graph $K_{3,3}$ (see [19]) as a subgraph. However, there are non-planar graphs having dimension 2. The dimension of an arbitrary graph with n nodes can be approximated within a factor $O(\log n)$ in polynomial time. Finally, finding the metric dimension of an arbitrary graph is NP-hard.

In [5], Chartrand *et al.* presented bounds of the dimension of a graph G in terms of the order and the diameter of G , and determined all connected graphs of order n having dimension $n-1$ or $n-2$. They also showed how to find the metric dimension and a basis for a graph in terms of an integer programming problem. A collection of bounds or exact values of metric dimensions is summarized in [11], for several families of connected graphs as well as for the join and the Cartesian product of graphs. The dimension of a graph with an added vertex is studied in [3]. For results on infinite locally finite graphs, see [4].

In digital geometry, the notion of metric dimension is equally natural. Melter and Tomescu showed in [14] the following results. The metric bases for the digital plane with Euclidean distance d_2 consist precisely of sets of three non-collinear points, whereas with respect to the city block distance d_1 and the chessboard distance d_∞ , the digital plane has no finite metric bases. From the point of view of applications, only finite sets are of interest, mainly rectangular regions. In the case of axes-parallel rectangles, the d_1 metric dimension of a rectangle is 2, and the d_∞ metric dimension of a square is 3. If non axes-parallel rectangles are considered, the situation is less simple: for both distances, there exists a rectangle in the digital plane such that its dimension is n , for any given $n \geq 3$. In [13], Khuller *et al.* proved that the d_1 metric dimension of a n -dimensional axes-parallel rectangle is n .

Several refinements of these bases notions have been proposed. Chartrand and Zhang defined and studied in [8] the notions of forcing subset, number and dimension. A subset F of a basis S of W is a *forcing subset* of S if S is the unique basis of W containing F . The *forcing number* is the minimum cardinality of a forcing subset for S , while the *forcing dimension* is the minimum forcing number among all bases of W . The forcing concepts have been studied for various subjects, see [8] for references.

Another interesting notion is the partition dimension, proposed in [6][7]. Let $\Pi = (S_1, \dots, S_k)$ be an ordered k -partition of W , and consider the distance to a set $d(p, S) = \min\{d(p, q) : q \in S\}$ where $p \in W$ and $S \subset W$. The representation of p with respect to Π is $r(p|\Pi) = \{d(p, S_1), \dots, d(p, S_k)\}$. The partition dimension $\text{pd}(W)$ and partition bases are then defined in the same manner as the metric ones. It is shown that for any nontrivial connected graph G we have $\text{pd}(G) \leq \dim(G) + 1$ [6]. Tomescu showed in [18] that the partition dimension may be much smaller than the metric dimension. In particular, the partition dimension of the digital plane with respect to d_1 and d_∞ is 3 and 4, respectively.

These concepts have important applications. In fact, distance geometry has immediate relevance where distance values are considered, such as in cartography, physics, biology, chemistry, classification, combinatorial search and optimization, game theory, and image analysis. For instance, locating and reference

sets are useful when working with sonar and LORAN stations for navigation [16]. In robot motion planing, minimizing a set of landmarks uniquely determine a robot's position [13]. A chemical compound is frequently viewed as a labeled graph, for which a metric basis allows to determine functional groups [3].

We are here mainly concerned by image analysis and digital geometry. While the metric bases and dimension concepts have been studied in [14] for d_1 and d_∞ , there is up to our knowledge no study available for an important class of digital metrics, widely used in image analysis: the chamfer distances. We aim at studying these metric concepts for these distances, and more generally for the class of polyhedral distance gauges, coming from Euclidean geometry.

The paper is organized as follows. In section 2 we recall the classical definitions of metric, norm and gauge, we state additional properties of gauges, and describe the link between gauges and chamfer norms. In section 3, we show that polyhedral gauges do not have finite metric bases in \mathbb{R}^n . Then, in section 4, we take a look at some classes of gauges having small metric dimension in a rectangle. Next in section 5, we present some experimental results. We finally conclude in section 6.

2 Norms and Gauges

2.1 Preliminaries

We first recall some classical definitions. A *metric* d on a nonempty set W is a function $d : W \times W \rightarrow \mathbb{R}$ which is positive defined, symmetric and sub-additive (the triangle inequality holds). A set W associated to a metric d is called a *metric space*, denoted by (W, d) .

Given a metric space (W, d) , the (closed) *ball* of center $p \in W$ and radius $r \in \mathbb{R}$ is the set $\mathcal{B}(p, r) = \{q \in W : d(p, q) \leq r\}$.

To define a norm, we need the notion of vector space, which we don't have in a graph, but is natural in \mathbb{R}^n and can be extended in \mathbb{Z}^n . A *vector space* is a set of vectors defined over a field. The generalization of the notion of a vector space over a ring A for a set W is named a *module*, denoted (W, A) . Each module can be associated to an affine space, and reciprocally. Since \mathbb{Z}^n is a module over \mathbb{Z} , this concept allows us to properly define a norm on \mathbb{Z}^n .

A *norm* g on a module (W, A) with values in \mathbb{R} is a function $g : W \rightarrow \mathbb{R}$ which is positive defined, homogeneous in A and sub-additive. Each norm induces a metric. A metric d is associated to a norm g if and only if d is translation invariant and homogeneous in A .

The Minkowski distance of order $p \geq 1$ between two points $x = (x_1, \dots, x_n)$ and $y = (y_1, \dots, y_n)$ is defined by $d_p(x, y) = (\sum_{i=1}^n |x_i - y_i|^p)^{1/p}$. It is well-known that any Minkowski distance is a metric and induces a norm (denoted ℓ_p). The d_1 distance is also called Manhattan or city block distance, the d_∞ distance is known as the chessboard distance, and d_2 is the Euclidean distance.

We call Σ^n the group of axial and diagonal symmetries in \mathbb{Z}^n about centre O . The cardinal of the group is $\#\Sigma^n = 2^n n!$ (which is 8, 48 and 384 for $n = 2, 3$ and 4). A set S is said to be *grid-symmetric* if for every $\sigma \in \Sigma^n$ we have

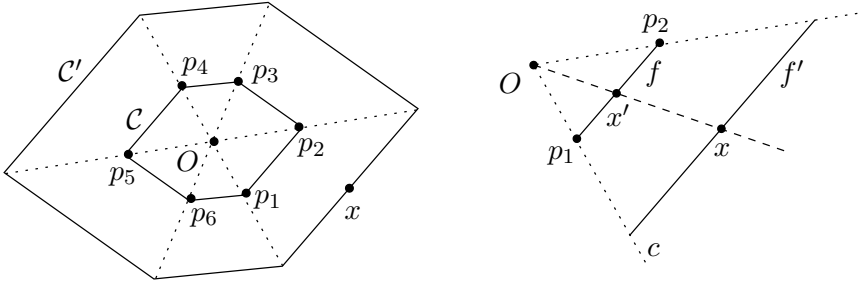


Fig. 1. Left: A gauge γ_C , its unit ball C , and the convex ball C' scaled by $d_C(O, x)$. Right: Illustration of the proof of lemma 1.1

$\sigma(S) = S$. The generator of a grid-symmetric set $S \subseteq \mathbb{Z}^n$ (or \mathbb{R}^n) is $\mathcal{G}(S) = \{(p_1, \dots, p_n) \in S : 0 \leq p_n \leq p_{n-1} \leq \dots \leq p_1\}$. The Minkowski distance balls are grid-symmetric.

2.2 Gauges

Gauges are usually defined in the Euclidean space \mathbb{R}^n and can be considered in the digital space \mathbb{Z}^n as well.

Given a convex C containing the origin O in its interior, a gauge for C is the function $\gamma_C(x)$ defined by the minimum positive scale factor λ , necessary for having $x \in \lambda C$. Formally, $\gamma_C(x) = \inf\{\lambda \in \mathbb{R}_+ : x \in \lambda C\}$. By construction, this function is positive, homogeneous in \mathbb{R} and sub-additive.

By definition, all norms are gauges for their unit ball. Conversely, a gauge for C is a norm, denoted n_C , iff C is central-symmetric [1]. We call distance gauge, denoted d_C , the metric induced by a central-symmetric gauge γ_C .

Since we are concerned with norms, we will only consider central-symmetric gauges in the remainder of the paper.

We now deal with the case where C is a central-symmetric convex polyhedron centered in O . Pick a facet f of C and denote p_1, \dots, p_m its vertices. Let x be a point of the cone $c = (O, f)$ (see Fig. 1.1). We can express x as a unique positive linear combination of the vertices: $\exists \lambda_1, \dots, \lambda_m \in \mathbb{R}_+$ such that $x = \sum_{i=1}^m \lambda_i p_i$.

Lemma 1 (Polyhedral gauge distance). $d_C(O, x) = \sum_{i=1}^m \lambda_i$.

Proof. Let $x \in C \setminus \{O\}$. There exists $\{\lambda_i\}_{1 \leq i \leq m}$ such that $x = \sum_i \lambda_i p_i$ and $\forall i, \lambda_i \geq 0$. Fix $x' = f \cap [Ox]$. Then $\exists \lambda > 0$ such that $x = \lambda x'$. Hence $x' = \frac{1}{\lambda} \sum_i \lambda_i p_i = \sum_i \frac{\lambda_i}{\lambda} p_i$. Since $x' \in f$, we have $\sum_i \frac{\lambda_i}{\lambda} = 1$. Thus $\lambda = \sum_i \lambda_i$. By definition $d_C(O, x) = \gamma_C(x) = \lambda \gamma_C(x')$ and $\gamma_C(x') = 1$, therefore $d_C(O, x) = \lambda$. \square

In the sequel, we denote by $g_c(x)$ the gauge distance $d_C(O, x)$ in the cone c .

Lemma 2 (Local additivity). $\forall x, y \in c, g_c(x + y) = g_c(x) + g_c(y)$.

Proof. By definition of the cone c we have $c = \{ \sum_i \delta_i p_i : \delta_i \in \mathbb{R}_+ \}$. Let $x = \sum_i \alpha_i p_i$, $\alpha_i \in \mathbb{R}_+$ and $y = \sum_i \beta_i p_i$, $\beta_i \in \mathbb{R}_+$. Then $x+y = \sum_i (\alpha_i p_i) + \sum_i (\beta_i p_i) = \sum_i (\alpha_i + \beta_i) p_i$. Since $\alpha_i + \beta_i \geq 0$ then $x + y \in c$. By application of lemma [11](#) we have $g_c(x) = \sum_i \alpha_i$, $g_c(y) = \sum_i \beta_i$ and $g_c(x + y) = \sum_i \alpha_i + \beta_i$. \square

2.3 Chamfer Norms

The chamfer distances have been extensively used and studied in image analysis, see [17](#) [12](#) [15](#) for references. The d_1 and d_∞ distances are peculiar cases of chamfer distances, and the class of chamfer norms can be seen as a special case of polyhedral gauges.

Here we recall some results from [17](#) §4.2-4.3]. A *chamfer mask* \mathcal{M} in \mathbb{Z}^n is a central-symmetric set $\mathcal{M} = \{ (\vec{v}_i, w_i) \in \mathbb{Z}^n \times \mathbb{Z}_{+*} \}_{1 \leq i \leq m}$ containing at least a basis of \mathbb{Z}^n , where (\vec{v}_i, w_i) are called *weightings*, \vec{v}_i *vectors* and w_i *weights*. The *chamfer distance* $d_{\mathcal{M}}$ between two points $p, q \in \mathbb{Z}^n$ is

$$d_{\mathcal{M}}(p, q) = \min \left\{ \sum \lambda_i w_i : \sum \lambda_i \vec{v}_i = \vec{p}q, 1 \leq i \leq m, \lambda_i \in \mathbb{Z}_+ \right\}. \quad (1)$$

If we consider the infinite weighted graph $G_{\mathcal{M}}$ where the vertices are the points of \mathbb{Z}^n , and the edges and weights are given by the weightings of \mathcal{M} translated around each point, then $d_{\mathcal{M}}$ is the intrinsic distance of $G_{\mathcal{M}}$, hence $d_{\mathcal{M}}$ is a metric.

Let $\mathcal{M}' = \{ O + \vec{v}_i/w_i \}_{1 \leq i \leq m} \in \mathbb{R}^n$ and let $B'_{\mathcal{M}} = \text{conv}(\mathcal{M}')$, then $B'_{\mathcal{M}}$ is a central-symmetric and convex polyhedron whose facets separate \mathbb{R}^n in cones from O . A facet \mathcal{F} of $B'_{\mathcal{M}}$ is generated by a subset $\mathcal{M}|_{\mathcal{F}} = \{ (\vec{v}_j, w_j) \}$ of \mathcal{M} ; if \mathcal{F} is simplicial and if $\Delta_{\mathcal{F}} = \det \{ \vec{v}_j \}$ is such that $|\Delta_{\mathcal{F}}| = 1$, then \mathcal{F} is said *unimodular*.

In the Euclidean space \mathbb{R}^n , we can define an analytic continuation $d_{\mathcal{M}}^{\mathbb{R}}$ of $d_{\mathcal{M}}$ by replacing $\lambda_i \in \mathbb{Z}_+$ with $\lambda_i \in \mathbb{R}_+$ in [\(1\)](#). It is easy to see that $d_{\mathcal{M}}^{\mathbb{R}}$ is the distance gauge for $B'_{\mathcal{M}}$, thus $d_{\mathcal{M}}^{\mathbb{R}}$ is always a norm; while in \mathbb{Z}^n , the following norm condition has to be fulfilled: $d_{\mathcal{M}}$ is a norm in \mathbb{Z}^n if and only there exists a unimodular triangulation of the facets of $B'_{\mathcal{M}}$ (see [17](#) p. 53] and [12](#) §6.3], or an equivalent condition in [15](#) §3.3]). This condition guaranties that $d_{\mathcal{M}}$ in \mathbb{Z}^n is the Gauss discretization of $d_{\mathcal{M}}^{\mathbb{R}}$ in \mathbb{R}^n .

Now let $d_{\mathcal{M}}$ be a chamfer norm, \mathcal{F} a simplicial facet of $B'_{\mathcal{M}}$ and $\mathcal{M}|_{\mathcal{F}} = \{ (\vec{v}_j, w_j) \}_{1 \leq j \leq n}$; then for each point $p = (p_1, \dots, p_n)$ in the cone (O, \mathcal{F}) , called *influence cone* of $\mathcal{M}|_{\mathcal{F}}$, we have $d_{\mathcal{M}}(O, p) = p_1 \delta_1 + \dots + p_n \delta_n$, where $(\delta_1, \dots, \delta_n) = \vec{\delta}_{\mathcal{F}}$ is a normal vector of \mathcal{F} , and δ_k is the *elementary displacement* for the k^{th} coordinate:

$$\delta_k = \frac{(-1)^{n+k}}{\Delta_{\mathcal{F}}} \cdot \begin{vmatrix} v_{1,1} & \cdots & v_{1,k-1} & v_{1,k+1} & \cdots & v_{1,n} & w_1 \\ \vdots & & \vdots & \vdots & & \vdots & \vdots \\ v_{n,1} & \cdots & v_{n,k-1} & v_{n,k+1} & \cdots & v_{n,n} & w_n \end{vmatrix}^T. \quad (2)$$

3 Metrics Bases for Polyhedral Gauges in Infinite Space

We show in this section that polyhedral gauges do not have finite bases in \mathbb{R}^n .

Given a convex polyhedron \mathcal{C} in \mathbb{R}^n , an hyperplane h_i is a *supporting hyperplane* if h_i contains a facet f_i of \mathcal{C} . We name *supporting half-space* H_i the half-space bounded by h_i and containing \mathcal{C} . The intersection $\cap_i H_i$ of the supporting half-spaces is equal to \mathcal{C} . In \mathbb{R}^n , the decomposition of \mathcal{C} in supporting half-spaces is unique. Note that this is generally not the case in \mathbb{Z}^n , see [15].

Let $\{H_i\}$ be the representation in half-spaces of a convex polyhedron $\mathcal{C} \in \mathbb{R}^n$ having nonempty interior, and $\{h_i\}$ the corresponding supporting hyperplanes. If the intersection $\cap_i h_i$ is a single point a , then \mathcal{C} is a *polyhedral cone* and a is called its *apex*. For convenience, this cone is denoted by $(a, \{H_i\})$. By construction, a polyhedral cone is unbounded. We name Λ the set of the polyhedral cones in \mathbb{R}^n having non-empty interior. A *ray* stands for a half-line.

Lemma 3. *For each cone $c = (a, \{H_i\}) \in \Lambda$, it always exists a point $p \in c$ and a ray $]ap) \subset c$ that does not belong to any facet of c .*

Proof. c is nonempty by definition of Λ , so an interior point p always exists. Let us prove that $p \notin \cup_i h_i \Rightarrow \forall \lambda > 0, a + \lambda p \notin \cup_i h_i$. Suppose that $\exists \lambda_0 > 0$ such that $q = a + \lambda_0 p \in \cup_i h_i$. Then there is at least one i_0 such that $q \in h_{i_0}$, hence $(aq) \in h_{i_0}$, but $p \in (aq)$ so $p \in h_{i_0}$, a contradiction. \square

We now translate some supporting half-planes. Let us consider a cone $c = (a, \{H_i\}_{1 \leq i \leq m}) \in \Lambda$, and a set of translations $\{t_i\}_{1 \leq i \leq m}$.

Lemma 4. *The convex $c' = \cap \{H'_i = H_i + t_i\}_{1 \leq i \leq m}$ is an unbounded nonempty convex polyhedron.*

Proof. Given an interior point $p \in c$, we consider the line $L = (ap)$. For each $1 \leq i \leq m$, we have $p \notin h_i$ so L intersects h_i in the single point a . Since $p \in H_i$, $L \cap H_i$ is the ray $]ap)$. Using the same argument, we deduce that $L \cap H'_i$ is a ray. Since $c' = \cap_i H'_i$ we have $c' \cap L = (\cap_i H'_i) \cap L = \cap_i (H'_i \cap L)$, thus $c' \cap L$ is the intersection of rays belonging to the same line L , having same orientation but having different apexes, so $c' \cap L$ is a ray, and we can conclude that c' is unbounded and nonempty. \square

It is easy to see that the resulting convex is not necessarily a cone.

We have just considered a single cone and we have applied a number of translations of the half-spaces defining the cone. We will now see a property concerning several cones obtained by translations on a common starting cone. Given a cone $c = (a, \{H_i\}_{1 \leq i \leq m}) \in \Lambda$ and a set $\{t_j\}_{1 \leq j \leq k}$ of translations in \mathbb{R}^n , we define the cones c_1, \dots, c_k as $c_j = c + t_j = (a + t_j, \{H_i + t_j\}_{1 \leq i \leq m})$.

Lemma 5. *The intersection of the cones c_1 to c_k is an unbounded and nonempty convex polyhedron.*

Proof. The intersection $\cap_{j=1}^k (c_j)$ is equal to $\cap_{j=1}^k (\cap_{i=1}^m (H_i + t_j))$, which is equal to $\cap_{i=1}^m (\cap_{j=1}^k (H_i + t_j))$. Now it is self-evident that each $\cap_{j=1}^k (H_i + t_j)$ results in a single half-space (by intersection of parallel half-spaces). More precisely, $\cap_{j=1}^k (H_i + t_j) = H_i + t'_i$ where $t'_i \in \{t_1, \dots, t_k\}$. So we can rewrite $\cap_{j=1}^k c_j = \cap_{i=1}^m (H_i + t'_i)$. The result follows by lemma 4. \square

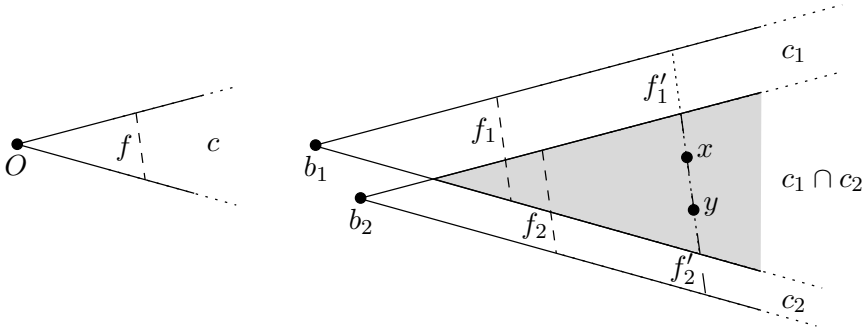


Fig. 2. On the left, the cone $c = (O, f)$. On the right, the two cones defined by $c_1 = c + \overrightarrow{Ob_1}$ and $c_2 = c + \overrightarrow{Ob_2}$. In grey the intersection between c_1 and c_2 . As a result, x and y both lie on f'_1 and f'_2 .

The lemma [1](#) provides a simple formula to calculate the distance values in a polyhedral gauge. So as to compute the formula, we need to split the space into cones. Given a polyhedral gauge γ_C and a facet f of \mathcal{C} , consider the cone $c = (O, f)$. Fix b_1 and b_2 two distinct points. Denote $c_1 = c + \overrightarrow{Ob_1}$ and $c_2 = c + \overrightarrow{Ob_2}$ the translated cones respectively centered in b_1 and b_2 (see Fig. [2](#)).

Lemma 6. $\forall x, y \in c_1 \cap c_2, g_{c_1}(x) = g_{c_2}(y) \iff g_{c_2}(x) = g_{c_1}(y)$.

Proof. Given a cone $c = (O, f)$ and two points b_1 and b_2 , we define $c_1 = c + \overrightarrow{Ob_1} = (b_1, f_1)$ and $c_2 = c + \overrightarrow{Ob_2} = (b_2, f_2)$. Set x a point in $c_1 \cap c_2$. We name f'_1 the facet f_1 scaled by $\lambda_1 = d_C(b_1, x) = g_{c_1}(x)$ and name f'_2 the facet f_2 scaled by $\lambda_2 = d_C(b_2, x) = g_{c_2}(x)$. Since f'_1 and f'_2 are parallel and intersect x , f'_1 and f'_2 are in the same hyperplane. Hence, for any point $y \in f'_1$, we also have $y \in f'_2$. Finally, by definition of gauges, we have $g_{c_2}(x) = g_{c_2}(y)$. \square

Lemma [6](#) also holds with several points. Indeed, considering any point b_3 , denote c_3 the cone defined by $c_3 = c + \overrightarrow{Ob_3}$. Then, by transitivity, $\forall x, y \in \cap_i c_i, g_{c_1}(x) = g_{c_1}(y) \iff g_{c_2}(x) = g_{c_2}(y) \iff g_{c_3}(x) = g_{c_3}(y)$.

Theorem 1. *There are no finite metric bases for polyhedral gauges in \mathbb{R}^n .*

Proof. Let us consider a polyhedral gauge γ_C . Suppose that $B = \{b_1, \dots, b_k\}$ is a metric basis for (\mathbb{R}^n, d_C) . Considering a cone $c = (O, f)$ defined by a facet f of \mathcal{C} and the origin O , we denote c_1, \dots, c_k the cones defined by $c_i = c + \overrightarrow{Ob_i}$. By lemma [5](#), we know that $\cap_{i=1}^k c_i$ is nonempty and unbounded. Therefore, there exist two points $x, y \in \cap_i c_i$ such that $g_{c_1}(x) = g_{c_1}(y)$. By lemma [6](#), x and y have the same representation $r(x|B) = r(y|B)$, thus B is not a resolving set for \mathbb{R}^n . \square

This theorem can be extended to any central-symmetric gauge partially polyhedral. Indeed, the existence of one hyperplanar facet is sufficient to define a cone in which no metric basis exists.

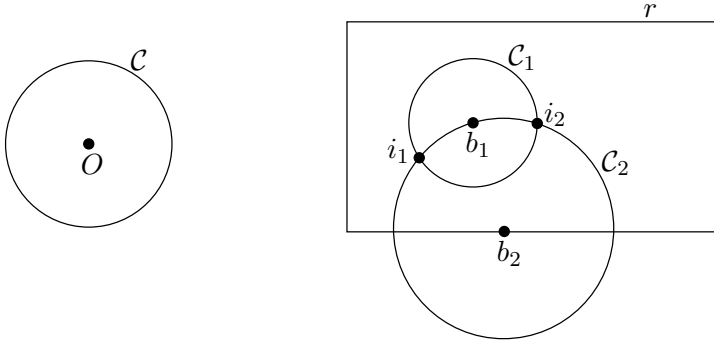


Fig. 3. Illustration for the proof of lemma 7. Left: the convex C defining the gauge γ_C . Right: the balls C_1 and C_2 centered in b_1 and b_2 intersect one another in i_1 and i_2 inside the rectangle r .

Having at least two points which have the same distance from O in a cone is a necessary condition for the theorem 4. This is not the case for all polyhedral gauges in \mathbb{Z}^n ; indeed, if for any radius, each facet intersects a single point of \mathbb{Z}^n then the metric dimension may be finite. This case may be detailed in an extended version of this paper. This condition is never met for chamfer norms; indeed, in each simplicial cone for a sufficiently large radius, the facet will intersect at least two points of \mathbb{Z}^n , because its normal has rational coordinates. Hence the theorem 4 remains valid for chamfer norms in \mathbb{Z}^n .

4 Metric Bases for Gauges in Axes-Parallel Rectangles

We consider here either polyhedral and non-polyhedral gauges. Let γ_C be a central-symmetric gauge and r be an axes-parallel rectangle in \mathbb{R}^2 .

Lemma 7. *If $B = \{b_1, b_2\}$ is a metric basis for r , then b_1 and b_2 are points of the frontier of r .*

Proof. Consider $B = \{b_1, b_2\}$ a metric basis for r , and suppose that b_1 is an interior point of r . Then we fix λ_1 such that $C_1 = (\lambda_1 C + \overrightarrow{Ob_1}) \in r$ and $b_2 \notin C_1$. Let $\lambda_2 = d_C(b_1, b_2)$ and $C_2 = (\lambda_2 C + \overrightarrow{Ob_2})$. Hence the intersection $\partial(C_1) \cap \partial(C_2)$, where $\partial(A)$ is the border of A , results in two distinct points i_1 and i_2 in r . Finally $r(i_1|B) = r(i_2|B)$, so B is not a resolving set for (r, d_C) . □

The lemma 7 is not valid in \mathbb{Z}^n . In fact, the intersection of discrete balls in \mathbb{Z}^n might be slightly different from its continuous counterpart, resulting in less or more than two points. See section 5 for examples.

Lemma 8. *Suppose that γ_C is a grid-symmetric gauge. If C does not contains any vertical nor horizontal facet, then the metric dimension of (r, d_C) is 2.*

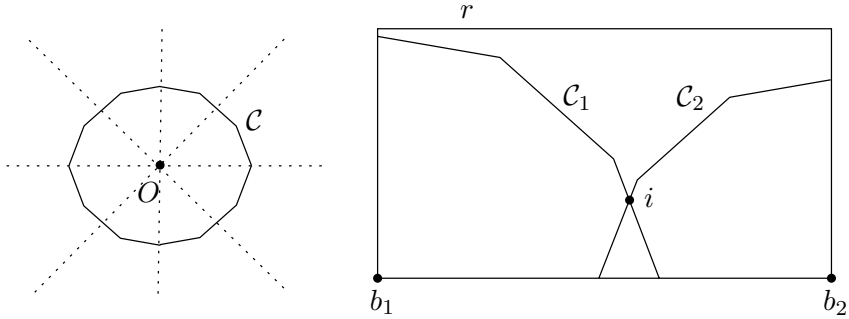


Fig. 4. Illustration of lemma 8. Left: a grid-symmetric gauge and its unit ball \mathcal{C} . Right: a single intersection i between two balls \mathcal{C}_1 and \mathcal{C}_2 centered in b_1 and b_2 .

Proof. The dimension is clearly > 1 . Fix b_1 and b_2 such that b_1 is the bottom-left corner of r and b_2 is the bottom-right corner (See Fig. 4). For any $\lambda_1 > 0$, the intersection $(\lambda_1 \mathcal{C} + \overrightarrow{Ob_1}) \cap r$ is a monotonic decreasing curve, because the gauge is convex and grid-symmetric. For the same reasons, $\forall \lambda_2 > 0$, $(\lambda_2 \mathcal{C} + \overrightarrow{Ob_2}) \cap r$ is monotonic increasing. If we add the fact that \mathcal{C} does not contain any vertical or horizontal facet, then we have both strictly monotonic curves. Intersection between strictly monotonic increasing and decreasing curves is at most a single point, thus $\{b_1, b_2\}$ is a resolving set for $(r, d_{\mathcal{C}})$. \square

Corollary 1. *The metric dimension for all Minkowski distances except d_{∞} is 2 in a rectangle.*

Proof. d_{∞} is the only Minkowski distance whose balls contain vertical or horizontal facets. \square

Lemma 9. *Let $\gamma_{\mathcal{C}}$ be a polyhedral gauge of metric dimension 2 in a rectangle r , fix $\{b_1, b_2\}$ a metric basis of $(r, d_{\mathcal{C}})$ such that b_1 is not a vertex of r , and denote e_1 the edge of r containing b_1 . Then $b_2 \notin e_1$.*

Proof. A small enough factor $\lambda_1 > 0$ exists, such that \mathcal{C}_1 intersects e_1 in two points i_1 and i_2 , and such that $b_2 \notin [i_1, i_2]$. Suppose that $i_1 \in [b_2, i_2]$ and consider the cone c_1 defined by b_1 and a facet of \mathcal{C} which contains i_2 . Denote c_2 the cone centered in b_2 which contains the cone c_1 (see Fig. 5). So we have $\forall y, z \in c_1, g_{c_1}(y) = g_{c_1}(z) \Rightarrow g_{c_2}(y) = g_{c_2}(z)$. In consequence, B cannot be a metric basis in the rectangle. \square

5 Results and Discussion

We have developed a program which gives by enumeration all metric bases for any given chamfer norm in an axes-parallel rectangle. The figure 6 shows for d_1 and the chamfer norms $\langle 3, 4 \rangle$ and $\langle 5, 7, 11 \rangle$, the union of all metric bases vertices (in grey) in a 20×12 rectangle. As we can see, the vertices of a metric basis for

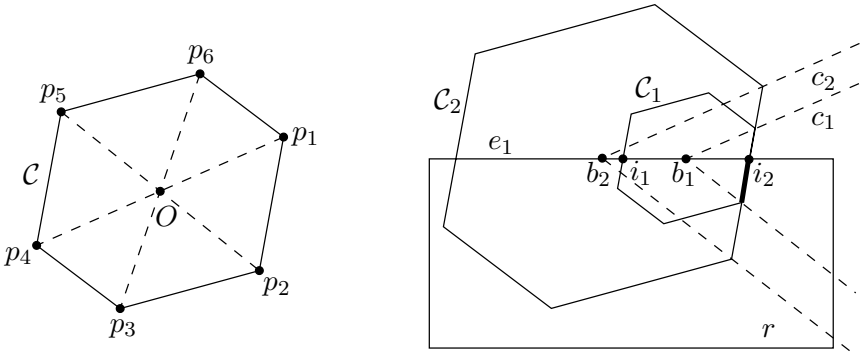


Fig. 5. Illustration of lemma 9. Left: A polyhedral gauge and its unit ball \mathcal{C} . Right: The bold line segment shows the intersection between the two balls \mathcal{C}_1 and \mathcal{C}_2 in r .

d_1 are necessarily located on the corners of the rectangle, while for $\langle 3, 4 \rangle$ they can be chosen in the first or second border inside the rectangle, and for the last case $\langle 5, 7, 11 \rangle$, in the whole rectangle. This shows that lemma 7 does not always hold in \mathbb{Z}^n .

We also developed an interactive program which displays, for a set S of vertices in a rectangular region W , the resolved and unresolved points (resp. in black, white and grey). Results are presented in Fig. 7; each column shows a configuration of two or three vertices; each row corresponds to a distance. The last distance is the chamfer norm $\langle 3, 4, 6 \rangle$, obtained by replacing the corresponding weights in the mask $\langle 5, 7, 11 \rangle$. The interest of this norm is that its balls are octogons with horizontal and vertical facets.

A point is said *resolved* in W if its representation for S is unique in W , while it is *unresolved* if there exists at least one other point in W sharing the same representation. A set S is then a resolving set for W if all points of W are resolved. In our example, only two of the 18 cases are resolving sets: rows (a) and (e) in the right column.

Now consider an axes-parallel subrectangle R in W , such that the black vertices lie on the corners of R . If all the points in R are resolved, then the set S of vertices is a resolving set for R . This is the case for (a,b,d,e)-right.

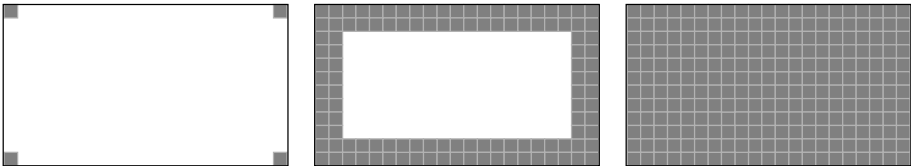


Fig. 6. Left to right: d_1 , $\langle 3, 4 \rangle$, $\langle 5, 7, 11 \rangle$

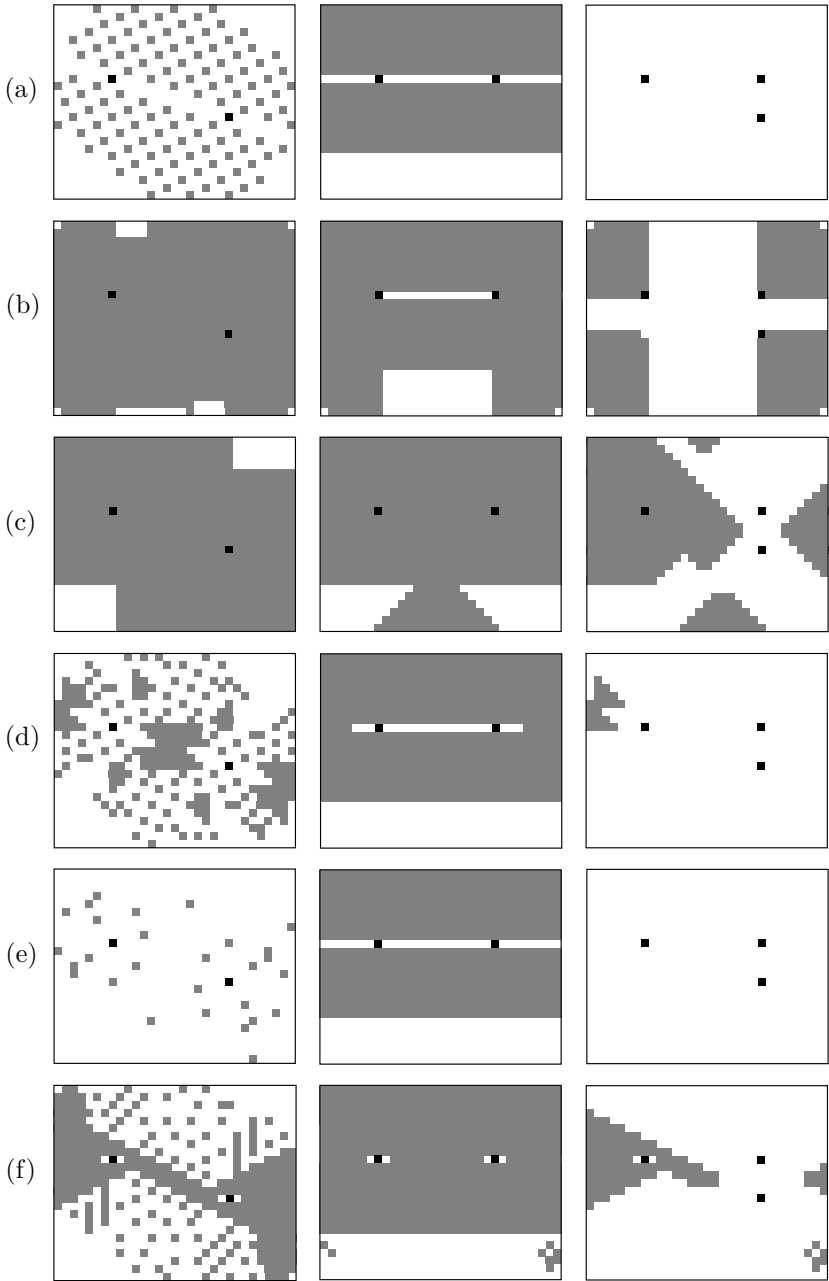


Fig. 7. Configurations for two or three vertices in a 31×25 rectangular region, using distances (a) d_2 , (b) d_1 , (c) d_∞ , (d) $\langle 3,4 \rangle$, (e) $\langle 5,7,11 \rangle$, (f) $\langle 3,4,6 \rangle$. The vertices are shown in black, resolved points in white, unresolved points in grey. The coordinates of vertices from the origin $(0,0)$ at top left are $(7,9)$, $(22,9)$ and $(22,14)$.

In some other cases, S is still a resolving set for R since each unresolved point in R has its equivalent points (points sharing the same representation, not represented here) lying outside R . We have this situation for (e)-left and (a,b,d,e)-middle, but not for (c,f)-middle. This can be explained by lemma [8](#), since the balls of d_∞ and $(3, 4, 6)$ have horizontal and vertical facets.

These different configurations are resulting from intersections of the cones in the distance balls. A thorough geometrical and arithmetical study might be a natural continuation of this paper for better understanding.

6 Conclusion

We have shown that the metric dimension of any polyhedral (or partially polyhedral) central symmetric gauge is infinite in \mathbb{R}^n , and for \mathbb{Z}^n in the case of chamfer norms, whereas it is finite in axes-parallel rectangles. In the latter case, we have exhibited gauges having metric dimension 2 and we have completely characterized their metric bases. In the future, we aim at studying how to generalize our results for continuous and discrete gauges in rectangles, in convex or non-convex simple polyhedrons with direct or geodesic distances; showing the conditions where discrete polyhedral gauges may have finite basis in \mathbb{Z}^n , and finally, tackling the linked problems of forcing subsets and partitions dimension.

References

1. Berger, M.: *Géométrie*. Nathan (1990)
2. Blumenthal, L.M.: *Theory and applications of distance geometry*. Chelsea publishing company, New York (1970)
3. Buczkowski, P.S., Chartrand, G., Poisson, C., Zhang, P.: On k -dimensional graphs and their bases. *Periodica Mathematica Hungarica* 46, 9–15 (2003)
4. Cáceres, J., Hernando, C., Mora, M., Pelayo, I.M., Puertas, M.L.: On the metric dimension of infinite graphs. *Electronic Notes in Discrete Math.* 35, 15–20 (2009)
5. Chartrand, G., Eroh, L., Johnson, M.A., Oellermann, O.R.: Resolvability in graphs and the metric dimension of a graph. *Discrete Applied Math.* 105(1-3), 99–113 (2000)
6. Chartrand, G., Salehi, E., Zhang, P.: On the partition dimension of a graph. *Congressus Numerantium* 131, 55–66 (1998)
7. Chartrand, G., Salehi, E., Zhang, P.: The partition dimension of a graph. *Aequationes Mathematicae* 59, 45–54 (2000)
8. Chartrand, G., Zhang, P.: The forcing dimension of a graph. *Mathematica Bohemica* 126(4), 711–720 (2001)
9. Harary, F., Melter, R.A.: On the metric dimension of a graph. *Ars Combinatoria* 2, 191–195 (1976)
10. Hardy, G.H., Wright, E.M.: *An introduction to the theory of numbers*, 5th edn. Oxford Science Pub. (1979)
11. Hernando, C., Mora, M., Pelayo, I.M., Seara, C., Cáceres, J., Puertas, M.L.: On the metric dimension of some families of graphs. *Electronic Notes in Discrete Mathematics* 22, 129–133 (2005)

12. Hulin, J.: Axe médian discret: propriétés arithmétiques et algorithmes. Thèse de Doctorat, Aix-Marseille Université (November 2009), <http://pageperso.lif.univ-mrs.fr/~jerome.hulin/these-hulin.pdf>
13. Khuller, S., Raghavachari, B., Rosenfeld, A.: Landmarks in graphs. *Discrete Applied Mathematics* 70(3), 217–229 (1996)
14. Melter, R.A., Tomescu, I.: Metric bases in digital geometry. *Computer Vision, Graphics, and Image Processing* 25(1), 113–121 (1984)
15. Normand, N., Evenou, P.: Medial axis lookup table and test neighborhood computation for 3D chamfer norms. *Pattern Recognition* 42(10), 2288–2296 (2009)
16. Slater, P.J.: Leaves of trees. *Congressus Numerantium* 14, 549–559 (1975)
17. Thiel, E.: Géométrie des distances de chanfrein. Habilitation à Diriger des Recherches, Université de la Méditerranée, Aix-Marseille 2 (December 2001), <http://pageperso.lif.univ-mrs.fr/~edouard.thiel/hdr>
18. Tomescu, I.: Discrepancies between metric dimension and partition dimension of a connected graph. *Discrete Mathematics* 308(22), 5026–5031 (2008)
19. Wikipedia: Gallery of named graphs, http://en.wikipedia.org/wiki/Gallery_of_named_graphs

Completions and Simplicial Complexes

Gilles Bertrand

Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge
Equipe A3SI, ESIEE Paris

Abstract. In this paper, we first introduce the notion of a completion. Completions are inductive properties which may be expressed in a declarative way and which may be combined. In the sequel of the paper, we show that completions may be used for describing structures or transformations which appear in combinatorial topology. We present two completions, $\langle C_{UP} \rangle$ and $\langle C_{AP} \rangle$, in order to define, in an axiomatic way, a remarkable collection of acyclic complexes. We give few basic properties of this collection. Then, we present a theorem which shows the equivalence between this collection and the collection made of all simply contractible simplicial complexes.

Keywords: Completions, simplicial complexes, collapse, simple homotopy, combinatorial topology.

1 Introduction

Several approaches have been proposed for the study of topological properties of digital objects in the context of computer imagery:

- The digital topology approach introduced by A. Rosenfeld [5]. Elements of \mathbf{Z}^d are linked by some adjacency relations. It is not obvious, in this framework, to define certain topological notions (*e.g.*, a homotopy).
- The connected ordered topological space (COTS) approach introduced by E. Khalimsky [3]. The smallest neighborhood of each point of \mathbf{Z}^d differs from one point to another. This allows to recover the structure of a topology.
- The complex cellular approach. An object is seen as a structure consisting of elements of different dimensions called cells. As noticed by V. Kovalevsky [4], it is also possible, with this approach, to recover the structure of a topology.

The underlying topology in the last two approaches corresponds to an *Alexandroff space* [2]. An Alexandroff space is a topological space in which the intersection of any arbitrary family (not necessarily finite) of open sets is open. There is a deep link between Alexandroff spaces and preorders, *i.e.*, binary relations that are reflexive and transitive. To any Alexandroff space, we may associate a preorder \leq such that $x \leq y$ if and only if y is contained in all open sets that contain x . Conversely, a preorder determines an Alexandroff space: a set O is open for this space if and only if $x \in O$ and $x \leq y$ implies $y \in O$.

A map f between two preordered sets X and Y is monotone if $x \leq y$ in X implies $f(x) \leq f(y)$ in Y . We have the following result.

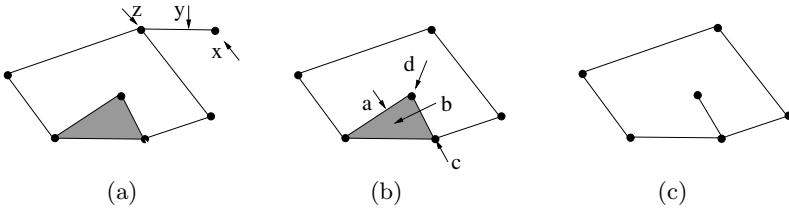


Fig. 1. (a): A simplicial complex X , (b): A complex $Y \subseteq X$, (c): A complex $Z \subseteq Y$.

A map between two preordered sets is monotone if and only if it is a continuous map between the corresponding Alexandroff spaces. Conversely, a map between two Alexandroff spaces is continuous if and only if it is a monotone map between the corresponding preordered sets.

Thus, there is a structural equivalence between Alexandroff spaces and preorders.

Let us consider the (simplicial) objects X, Y, Z depicted Fig. 1. The object X is made of 7 vertices, 8 segments, and 1 triangle. A natural preorder between all these elements is the partial order corresponding to the relation of inclusion between sets. Thus we have $x \leq y$ and $z \leq y$. Let the map f between X and Y be such that f is the identity on all elements of Y and $f(x) = z, f(y) = z$. We note that f is monotone, for example we have $x \leq y$ and $f(x) \leq f(y)$. Thus, Y may be seen as a continuous retraction of X for the corresponding topology. Now, let us try to build a monotone map g between Y and Z such that g is the identity on all elements of Z . We see that this is not possible. For example, if we take $g(a) = c, g(b) = c$, we have $d \leq a$, but we have not $g(d) \leq g(a)$. We also observe that it is possible to build such a map between Y and $Y' = Y \setminus \{a\}$, but not between Y' and Z . Thus, in the context of this construction, the classical axioms of topology fail to interpret Z as a continuous retraction of Y .

The paper is organized as follows. First we introduce the notion of a completion. Completions are inductive properties which may be expressed in a declarative way and which may be combined. In the sequel of the paper, we show that completions may be used for describing structures or transformations which appear in combinatorial topology. After some basic definitions for simplicial complexes, we give two examples of completions. We recall some definitions relative to the collapse operator which allows to make the two transforms illustrated Fig. 1. We present two completions, $\langle C_{UP} \rangle$ and $\langle C_{AP} \rangle$, in order to define a remarkable collection of acyclic complexes. We give few basic properties of this collection. Then, we present a theorem which shows the equivalence between this collection and the collection made of all simply contractible simplicial complexes.

The paper is self contained. For the sake of place, only few proofs are presented. The other one's will be given in an extended version of the paper.

2 Completions

We introduce the notions of a constructor and a completion. See also Appendix A for more details and for the link with finitary closure operators.

In the sequel, the symbol \mathbf{S} will denote an arbitrary collection. The symbol \mathcal{K} will denote an arbitrary subcollection of \mathbf{S} , thus we have $\mathcal{K} \subseteq \mathbf{S}$.

Definition 1. Let κ be a binary relation over $2^{\mathbf{S}}$ and $2^{\mathbf{S}}$, thus $\kappa \subseteq 2^{\mathbf{S}} \times 2^{\mathbf{S}}$. We say that κ is a constructor (on \mathbf{S}) if κ is finitary, which means that \mathbf{F} is finite whenever $(\mathbf{F}, \mathbf{G}) \in \kappa$. If κ is a constructor on \mathbf{S} , we denote by $\langle \kappa \rangle$ the following property which is the completion induced by κ :

$$\rightarrow \text{If } \mathbf{F} \subseteq \mathcal{K}, \text{ then } \mathbf{G} \subseteq \mathcal{K} \text{ whenever } (\mathbf{F}, \mathbf{G}) \in \kappa. \tag{\kappa}$$

Let κ be a constructor on \mathbf{S} and let $\mathbf{X} \subseteq \mathbf{S}$. We define:

$$\kappa(\mathbf{X}) = \cup\{\mathbf{G} \mid \text{there exists } (\mathbf{F}, \mathbf{G}) \in \kappa \text{ with } \mathbf{F} \subseteq \mathbf{X}\} \cup \mathbf{X}.$$

We set $\kappa^1(\mathbf{X}) = \kappa(\mathbf{X})$ and $\kappa^k(\mathbf{X}) = \kappa(\kappa^{k-1}(\mathbf{X}))$, $k \geq 2$.

We also set $\langle \mathbf{X}, \kappa \rangle = \cup\{\kappa^k(\mathbf{X}) \mid k \geq 1\}$.

Let $\langle K \rangle$ be a property which depends on \mathcal{K} . We say that a given collection $\mathbf{X} \subseteq \mathbf{S}$ satisfies $\langle K \rangle$ if the property $\langle K \rangle$ is true for $\mathcal{K} = \mathbf{X}$.

Theorem 1. Let κ be a constructor on \mathbf{S} and let $\mathbf{X} \subseteq \mathbf{S}$. There exists, under the subset ordering, a unique minimal collection which contains \mathbf{X} and which satisfies $\langle \kappa \rangle$, this collection is precisely $\langle \mathbf{X}, \kappa \rangle$.

Furthermore, we have $\langle \mathbf{X}, \kappa \rangle = \cap\{\mathbf{Y} \subseteq \mathbf{S} \mid \mathbf{X} \subseteq \mathbf{Y} \text{ and } \mathbf{Y} \text{ satisfies } \langle \kappa \rangle\}$.

We say that a property $\langle K \rangle$ is a *completion (property)* if there exists a constructor κ such that $\langle K \rangle = \langle \kappa \rangle$ which means that, for each $\mathcal{K} \subseteq \mathbf{S}$, $\langle K \rangle$ is true if and only if $\langle \kappa \rangle$ is true. If $\langle K \rangle$ is a completion and if $\mathbf{X} \subseteq \mathbf{S}$, we write $\langle \mathbf{X}, K \rangle$ for the unique minimal collection which contains \mathbf{X} and which satisfies $\langle K \rangle$.

Let $\langle K \rangle$ and $\langle Q \rangle$ be two completions induced, respectively, by the constructors κ and \mathbf{q} . We see that $\kappa \cup \mathbf{q}$ is a constructor and that $\langle K \rangle \wedge \langle Q \rangle$ is the property induced by $\kappa \cup \mathbf{q}$, the symbol \wedge standing for the logical “and”. Thus, if $\langle K \rangle$ and $\langle Q \rangle$ are completions, then $\langle K \rangle \wedge \langle Q \rangle$ is a completion.

In the sequel of the paper, we write $\langle K, Q \rangle$ for the completion $\langle K \rangle \wedge \langle Q \rangle$. Thus, if $\mathbf{X} \subseteq \mathbf{S}$, the notation $\langle \mathbf{X}, K, Q \rangle$ stands for the smallest collection which contains \mathbf{X} and which satisfies $\langle K \rangle \wedge \langle Q \rangle$.

Remark 1. We may build a counter-example which shows that, if $\langle K \rangle$ and $\langle Q \rangle$ are completions, then $\langle K \rangle \vee \langle Q \rangle$ is not necessarily a completion, the symbol \vee standing for the logical “or”.

A constructor κ is *one-to-one* if $Card(\mathbf{F}) = Card(\mathbf{G}) = 1$ whenever $(\mathbf{F}, \mathbf{G}) \in \kappa$. If κ is a one-to-one constructor, we set $-\kappa = \{(\mathbf{G}, \mathbf{F}) \mid (\mathbf{F}, \mathbf{G}) \in \kappa\}$, the constructor $-\kappa$ is the *inverse of κ* .

It may be seen that, if κ and \mathbf{q} are two one-to-one constructors, we have $\langle \kappa \rangle = \langle \mathbf{q} \rangle$ if and only if $\langle -\kappa \rangle = \langle -\mathbf{q} \rangle$.

A *one-to-one completion* $\langle K \rangle$ is a completion induced by some one-to-one constructor κ , we write $\langle -K \rangle$ for the constructor induced by $-\kappa$. By the preceding property, this definition is consistent since it does not depend on the choice of κ .

3 Basic Definitions for Simplicial Complexes

A *complex* or an *hypergraph* is a finite family composed of finite sets. We denote by \mathbb{H} the collection of all complexes.

Let $X \in \mathbb{H}$. The *simplicial closure* of X is the complex X^- such that $X^- = \{y \subseteq x \mid x \in X\}$. The complex X is a *simplicial complex* if $X = X^-$. We denote by \mathbb{S} the collection of all simplicial complexes. Observe that $\emptyset \in \mathbb{S}$ and $\{\emptyset\} \in \mathbb{S}$.

Let $X \in \mathbb{S}$. An element of X is a *simplex* of X or a *face* of X . A *facet* of X is a simplex of X which is maximal for inclusion.

A *simplicial subcomplex* of $X \in \mathbb{S}$ is any subset Y of X which is a simplicial complex. If Y is a subcomplex of X , we write $Y \preceq X$.

A complex $A \in \mathbb{S}$ is a *cell* if $A = \emptyset$ or if A has precisely one non-empty facet x , we set $A^\circ = A \setminus \{x\}$ and $\emptyset^\circ = \emptyset$. We write \mathbb{C} for the collection of all cells.

Let $X \in \mathbb{S}$. The *dimension* of $x \in X$, written $\dim(x)$, is the number of its elements minus one. The *dimension* of X , written $\dim(X)$, is the largest dimension of its simplices, the *dimension* of \emptyset is defined to be -1 .

If $\mathbb{X} \subseteq \mathbb{S}$, we set:

$$\mathbb{X}[d] = \{X \in \mathbb{X} \mid \dim(X) = d\} \text{ and } \mathbb{X}\langle d \rangle = \{X \in \mathbb{X} \mid \dim(X) \leq d\}.$$

The *ground set* of $X \in \mathbb{H}$ is the set $X^\dagger = \cup\{x \in X\}$. Let $X, Y \in \mathbb{H}$ such that $X^\dagger \cap Y^\dagger = \emptyset$. The *join* of X and Y is the complex XY such that $XY = \{x \cup y \mid x \in X, y \in Y\}$. Thus, $XY = \emptyset$ if $Y = \emptyset$ and $XY = X$ if $Y = \{\emptyset\}$.

In this paper, if $X, Y \in \mathbb{H}$, we implicitly assume that X and Y have disjoint ground sets whenever we write XY .

If $X \in \mathbb{S}$, we say that AX is a *simple cone* if $A \in \mathbb{C}[1]$, and a *cone* if $A \in \mathbb{C}$.

The (*reduced*) *Euler characteristic* of $X \in \mathbb{S}$ is the number $\chi(X)$ such that $\chi(X) = \sum\{(-1)^{\dim(x)} \mid x \in X\}$ if $X \neq \emptyset$, and $\chi(\emptyset) = 0$. Note that $\chi(X)$ is equal to the ordinary Euler characteristic minus one.

Let $A \in \mathbb{C}$ and $X \preceq A$. The *dual* of X for A is the simplicial complex, written $(X; A)^*$, such that $(X; A)^* = \{A^\dagger \setminus x \mid x \in A \setminus X\}$. Thus, we have $(X; A)^* = \{x \in A \mid (A^\dagger \setminus x) \notin X\}$. For any $A \in \mathbb{C}$, we have the following:

- If $X \preceq A$, then $((X; A)^*; A)^* = X$.
- If $X \preceq A$ and $Y \preceq A$, then $(X \cup Y; A)^* = (X; A)^* \cap (Y; A)^*$.
- If $X \preceq A$ and $Y \preceq A$, then $(X \cap Y; A)^* = (X; A)^* \cup (Y; A)^*$.
- We have $(\emptyset; A)^* = A$ and $(\{\emptyset\}; A)^* = A^\circ$.

In the sequel of this paper, we set $\mathbf{S} = \mathbb{S}$. Thus, we will have $\mathcal{K} \subseteq \mathbf{S}$.

In the next two sections, we give some basic examples of completions on \mathbf{S} .

4 Connectedness

The family composed of all connected simplicial complexes may be defined by means of completions. We define the one-to-one completion $\langle \text{PATH} \rangle$ as follows.

\rightarrow If $S \in \mathcal{K}$, then $S \cup C \in \mathcal{K}$ whenever $C \in \mathbb{C}$, and $S \cap C \neq \{\emptyset\}$. $\langle \text{PATH} \rangle$

We may easily verify that $\langle \text{PATH} \rangle$ is indeed a (one-to-one) completion. The property $\langle \text{PATH} \rangle$ is the completion induced by the (one-to-one) constructor:

$$\text{PATH} = \{(\{S\}, \{S \cup C\}) \mid S \in \mathbb{S}, C \in \mathbb{C} \text{ and } S \cap C \neq \{\emptyset\}\}.$$

We set $\Pi = \langle \emptyset, \text{PATH} \rangle$. We say that a complex $X \in \mathbb{S}$ is *connected* if $X \in \Pi$.

Observe that $\mathbb{C} \subseteq \Pi$ since, for any $C \in \mathbb{C}$, we have $C \cap \emptyset = \emptyset \neq \{\emptyset\}$. It may be checked that this definition of a connected complex is equivalent to the classical definition based on paths. Now let us define the completion $\langle \mathcal{Y} \rangle$ as follows.

\rightarrow If $S, T \in \mathcal{K}$, then $S \cup T \in \mathcal{K}$ whenever $S \cap T \neq \{\emptyset\}$. $\langle \mathcal{Y} \rangle$

Again, we may easily verify that $\langle \mathcal{Y} \rangle$ is indeed a completion. We have the following result which shows that \mathcal{Y} provides another way to generate Π .

Proposition 1. *We have $\Pi = \langle \mathbb{C}, \mathcal{Y} \rangle$*

A property similar to $\langle \mathcal{Y} \rangle$ has been introduced by J. Serra and G. Matheron who proposed, through the notion of a connection, a new set of axioms for connectedness [15]. The main difference between a connection and $\langle \mathbb{C}, \mathcal{Y} \rangle$ is that a connection may be seen as a “static structure” for modeling connectedness, on the contrary \mathcal{Y} is used in $\langle \mathbb{C}, \mathcal{Y} \rangle$ in a “dynamic way” for generating all elements of Π . On the other hand, \mathcal{Y} works only for finite objects.

5 Trees

A tree is classically defined as a graph which is path-connected and which does not contain any cycle. We give here a definition based on the following one-to-one completion $\langle \text{TREE} \rangle$.

\rightarrow If $S \in \mathcal{K}$, then $S \cup A \in \mathcal{K}$ whenever $A \in \mathbb{C}\langle 1 \rangle$ and $S \cap A \in \mathbb{C}\langle 0 \rangle$. $\langle \text{TREE} \rangle$

We set $\text{Tree} = \langle \emptyset, \text{TREE} \rangle$. We say that a complex $X \in \mathbb{S}$ is a *tree* if $X \in \text{Tree}$.

It may be checked that this definition of a tree is equivalent to the classical one.

Through this example, we observe that completions allow to define a collection in a constructive way rather by the means of properties of this collection.

We can express, in a concise manner, a fundamental property of trees.

Proposition 2. *We have $\text{Tree} = \langle \text{Tree}, -\text{TREE} \rangle$.*

6 Collapse

We now present some completions related to the collapse operator introduced by J.H.C. Whitehead [12]. Let us recall a classical definition of collapse.

Let $X \in \mathbb{S}$. We say that a face $x \in X$ is *free* for X if x is a proper face of exactly one face y of X , such a pair (x, y) is said to be a *free pair* for X . If (x, y) is a free pair for X , the complex $Y = X \setminus \{x, y\}$ is an *elementary collapse* of X .

We define the one-to-one completion $\langle \text{COL} \rangle$:

\rightarrow If $S \in \mathcal{K}$, then $S \cup AB \in \mathcal{K}$ whenever $A \in \mathbb{C}\langle 0 \rangle$, $AB \in \mathbb{C}$, $S \cap AB = AB^\circ$. $\langle \text{COL} \rangle$

Observe that, if $AB \in \mathbb{C}$, and if $A \in \mathbb{C}\langle 0 \rangle$, then necessarily $B \in \mathbb{C}$ or $B = \{\emptyset\}$. If $B = \{\emptyset\}$, B° is defined to be \emptyset .

It may be seen that, if S and $S \cup aB$, with $B \neq \emptyset$, fulfill the above conditions, then S is an elementary collapse of $S \cup aB$. Conversely, we may formulate any elementary collapse by such an expression. Thus $\langle \text{COL} \rangle$ is an alternative definition of collapse. The following is a direct consequence of the previous definitions.

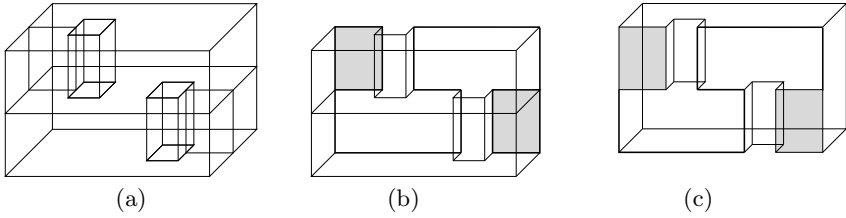


Fig. 2. (a): A Bing’s house X with two rooms, (b): An object $Y \subseteq X$, (c): An object $Z \subseteq X$. We have $X = Y \cup Z$, the object $Y \cap Z$ is outlined in (b) and (c).

Proposition 3. *Let $A \in \mathbb{C}$, $X \preceq A$, $Y \preceq A$. The complex X is an elementary collapse of Y if and only if $(Y; A)^*$ is an elementary collapse of $(X; A)^*$.*

We say that an element of $\langle \emptyset, \mathbf{COL} \rangle$ is *collapsible* and that an element of $\langle \emptyset, \mathbf{COL}, -\mathbf{COL} \rangle$ is *simply contractible*. If $X, Y \in \mathbb{S}$, we say that Y *collapses onto* X if $Y \in \langle \{X\}, \mathbf{COL} \rangle$, and that Y is *simple homotopic to* X if $Y \in \langle \{X\}, \mathbf{COL}, -\mathbf{COL} \rangle$.

Let us consider the one-to-one completion:

\rightarrow If $S \in \mathcal{K}$, then $S \cup A \in \mathcal{K}$ whenever $A \in \mathbb{C}$, and A collapses onto $S \cap A$. $\langle \mathbf{SIM} \rangle$

When A satisfies $\langle \mathbf{SIM} \rangle$, we say that the cell A is *simple for* S . We have $\langle \mathbf{SIM} \rangle = \langle \mathbf{COL} \rangle$. This completion leads to a notion of simplicity introduced in the context of computer imagery [16] (see also [17], [18]) where an object is often seen as a set of cells (*e.g.*, a set of *voxels* in 3D) rather than a set of faces.

7 The Cup/Cap Completions

We introduce the notion of a dendrite for defining a remarkable collection made of acyclic complexes.

Definition 2. *We define the two completions $\langle \mathbf{C}_{UP} \rangle$ and $\langle \mathbf{C}_{AP} \rangle$:*

\rightarrow If $S, T \in \mathcal{K}$, then $S \cup T \in \mathcal{K}$ whenever $S \cap T \in \mathcal{K}$. $\langle \mathbf{C}_{UP} \rangle$

\rightarrow If $S, T \in \mathcal{K}$, then $S \cap T \in \mathcal{K}$ whenever $S \cup T \in \mathcal{K}$. $\langle \mathbf{C}_{AP} \rangle$

We set $\mathbb{R} = \langle \mathbb{C}, \mathbf{C}_{UP} \rangle$ and $\mathbb{D} = \langle \mathbb{C}, \mathbf{C}_{UP}, \mathbf{C}_{AP} \rangle$.

Each element of \mathbb{R} is a ramification and each element of \mathbb{D} is a dendrite.

The Bing’s house with two rooms [13] is a classical example of an object which is contractible but not collapsible, this object is depicted Fig. 2(a). Let us consider the two complexes Y and Z of Fig. 2(b) and (c). They are such that $X = Y \cup Z$. If X is correctly triangulated, then Y, Z , and $Y \cap Z$ are ramifications. Thus, the Bing’s house X is a ramification.

It may be easily seen that we have $\langle \emptyset, \mathbf{COL} \rangle \subseteq \langle \mathbb{C}, \mathbf{C}_{UP} \rangle$. Since the Bing’s house is not collapsible, this inclusion is strict.

M. Hochster [10] (see also [8] [9]) introduced the notion of a constructible complex. This notion may be expressed using the following completion:

\rightarrow If $S, T \in \mathcal{K}[d]$, then $S \cup T \in \mathcal{K}$ whenever $S \cap T \in \mathcal{K}[d - 1]$, $d \geq 0$. $\langle \mathbf{CONS} \rangle$

A simplicial complex is *constructible* if it is an element of $\langle \mathbb{C} \cup \{\{\emptyset\}\}, \mathbf{CONS} \rangle$.

M. Hachimori [9], [11] showed that the Bing’s house with two rooms and the dunce hat [14] are not constructible. Observe that we have $\langle \mathbb{C}, \mathbb{C}_{\text{ONS}} \rangle \subseteq \langle \mathbb{C}, \mathbb{C}_{\text{UP}} \rangle$. Since the Bing’s house is a ramification, the previous inclusion is strict.

With the notion of a buildable complex, J. Jonsson [7] drops the condition for dimension which appears in $\langle \mathbb{C}_{\text{ONS}} \rangle$. The definition of a buildable complex is a recursive definition of what we call a ramification.¹ It was shown [7] that any buildable complex is contractible, *i.e.*, is homotopy equivalent to a single point. As far as we know, the above definition for dendrites has never been proposed.

8 Few Basic Properties

In this section, we give few basic properties which may be derived directly from the definitions of \mathbb{R} and \mathbb{D} using inductive arguments. Perhaps the simplest property which may be proved in such a way is the following.

Proposition 4. *If $X \in \mathbb{D}$, then $\chi(X) = 0$.*

Proof. We have $\chi(X) = 0$ for each $X \in \mathbb{C}$. Since the Euler characteristic is such that $\chi(S \cup T) = \chi(S) + \chi(T) - \chi(S \cap T)$, the result follows by induction. \square

Let us consider the two completions:

- \rightarrow If $S, T \in \mathcal{K}$, then $S \cup T \in \mathcal{K}$. $\langle \mathbb{U}_{\text{NION}} \rangle$
- \rightarrow If $S, T \in \mathcal{K}$, then $S \cap T \in \mathcal{K}$. $\langle \mathbb{I}_{\text{NTER}} \rangle$

Let $\mathbb{X} \subseteq \mathbb{R}$. An element of $\langle \mathbb{X}, \mathbb{U}_{\text{NION}} \rangle$ is, in general, not necessarily a ramification (nor a dendrite), but this property is true in the following case.

Proposition 5. *Let $\mathbb{X} \subseteq \mathbb{R}$ and let $\mathbb{Y} = \langle \mathbb{X}, \mathbb{U}_{\text{NION}} \rangle$. If $\langle \mathbb{Y}, \mathbb{I}_{\text{NTER}} \rangle = \mathbb{Y}$, *i.e.*, if \mathbb{Y} satisfies the property $\langle \mathbb{I}_{\text{NTER}} \rangle$, then we have $\mathbb{Y} \subseteq \mathbb{R}$.*

Proof. We set $\mathbb{Y}^k = \{X \in \mathbb{Y} \mid \text{Card}(X) \leq k\}$.

- i) We have $\mathbb{Y}^0 = \emptyset$ or $\mathbb{Y}^0 = \{\emptyset\}$. In both cases $\mathbb{Y}^0 \subseteq \mathbb{R}$.
- ii) Suppose $\mathbb{Y}^{k-1} \subseteq \mathbb{R}$, for some $k \geq 1$. Let $X \in \mathbb{Y}^k$. If $X \in \mathbb{X}$, then $X \in \mathbb{R}$. If $X \notin \mathbb{X}$, then there exists $S, T \in \mathbb{Y}$ such that $X = S \cup T$, and $S \not\subseteq T, T \not\subseteq S$. Thus $\text{Card}(S) \leq k-1, \text{Card}(T) \leq k-1$, and $\text{Card}(S \cap T) \leq k-1$. Furthermore, we have $S \cap T \in \mathbb{Y}$. Therefore, by the induction hypothesis, S, T and $S \cap T$ are ramifications, which means that X is a ramification. It follows that $\mathbb{Y}^k \subseteq \mathbb{R}$. \square

Let $A \in \mathbb{C}$ and let $\mathbb{X} = \{BA \mid B \in \mathbb{C}\}$. Since $XA \cup YA = (X \cup Y)A$, it may be seen that we have $\langle \mathbb{X}, \mathbb{U}_{\text{NION}} \rangle = \{XA \mid X \in \mathbb{S}\}$. Furthermore, since $XA \cap YA = (X \cap Y)A$, $\langle \mathbb{X}, \mathbb{U}_{\text{NION}} \rangle$ satisfies the property $\langle \mathbb{I}_{\text{NTER}} \rangle$. Thus, since $\mathbb{X} \subseteq \mathbb{R}$, the following is a consequence of Prop. 5.

Proposition 6. *Let $X \in \mathbb{S}$ and $A \in \mathbb{C}$. Then XA is a ramification.*

The following fact will be used for the proof of Prop. 8.

Proposition 7. *Let $A, B \in \mathbb{C}$. Then $(AB)^\circ = AB^\circ \cup A^\circ B$.*

¹ We suggest the name “ramification” rather than “buildable complex” since these objects may be seen as natural extensions of trees.

Proposition 8. *Let $X \in \mathbb{S}$ and $A \in \mathbb{C}$, $A \neq \emptyset$. The complex XA° is a dendrite if and only if X is a dendrite.*

Proof. If $A \in \mathbb{C}[0]$, we have $A^\circ = \{\emptyset\}$ and $XA^\circ = X$. Suppose the property is true for any $A \in \mathbb{C}\langle i-1 \rangle$, $i \geq 1$. Let $Y = XA^\circ$, with $X \in \mathbb{S}$ and $A \in \mathbb{C}[i]$. Since $i \geq 1$, there exists $B, C \in \mathbb{C}\langle i-1 \rangle$, such that $A = BC$. By Prop. 7, we have $XA^\circ = XBC^\circ \cup XB^\circ C$. By Prop. 6, XBC° and $XB^\circ C$ are dendrites. But $XBC^\circ \cap XB^\circ C = XB^\circ C^\circ$. By the induction hypothesis, $XB^\circ C^\circ$ is a dendrite iff XB° is a dendrite, and XB° is a dendrite iff X is a dendrite. Thus, by $\langle C_{UP} \rangle$ and $\langle C_{AP} \rangle$, Y is a dendrite iff X is a dendrite. \square

We now give a property for duality which will be used in the sequel through two corollaries.

Proposition 9. *Let $A, B \in \mathbb{C}$, and let $X, Y \in \mathbb{S}$ such that $X \preceq A$, $Y \preceq B$. We have $(XY; AB)^* = A(Y; B)^* \cup B(X; A)^*$.*

The following corollary may be obtained from Prop. 9, by interchanging A and B , and by setting $Y = A$. It shows that the collection of cones is closed by duality. More precisely, if we set $\mathbb{K} = \{AX \mid A \in \mathbb{C}, X \in \mathbb{S}\}$ and $\mathbb{K}^* = \{(X; B)^* \mid X \in \mathbb{K}, B \in \mathbb{C}, X \preceq B\}$, then we have $\mathbb{K} = \mathbb{K}^*$. Note that the collection \mathbb{C} is not closed by duality (By Cor. 2, we have $(A; AB)^* = B^\circ A$).

Corollary 1. *Let $A, B \in \mathbb{C}$, and let $X \preceq B$. We have $(AX; AB)^* = A(X; B)^*$.*

The second corollary may be obtained from Prop. 9 by setting $Y = \{\emptyset\}$.

Corollary 2. *Let $A, B \in \mathbb{C}$, and $X \preceq A$. We have $(X; AB)^* = B(X; A)^* \cup B^\circ A$.*

We see that this last formula allows to calculate the dual of an object in a given space (a cell) from the dual of this object in a smaller space.

Proposition 10. *Let $A, B \in \mathbb{C}$, and $X \preceq A$. Then $(X; AB)^*$ is a dendrite if and only if $(X; A)^*$ is a dendrite.*

Proof. Let $Y = (X; AB)^*$. By Cor. 2, we have $Y = B(X; A)^* \cup B^\circ A$. By Prop. 6, $B(X; A)^*$ and $B^\circ A$ are dendrites. Since $B(X; A)^* \cap B^\circ A = B^\circ(X; A)^*$ and by Prop. 8, it follows that Y is a dendrite if and only if $(X; A)^*$ a dendrite. \square

From the very definition of a dendrite, it is clear that, if X is a dendrite, then there exists $A \in \mathbb{C}$ such that $(X; A)^*$ is a dendrite. As a consequence of Prop. 10, we have the following which extends this property for all cells containing X .

Proposition 11. *If $X \in \mathbb{D}$ and if $X \preceq A$, with $A \in \mathbb{C}$, then $(X; A)^* \in \mathbb{D}$.*

9 Completions and Simple Homotopy

The following Th. 2 states the equivalence between dendrites and simply contractible complexes. For the sake of space, we give only one part (in fact the easiest part) of the proof.

Lemma 1. *Let $X \in \mathbb{S}$. If X is simply contractible, then X is a dendrite.*

Proof. Let $X \in \mathbb{S}$ such that X is simply contractible. Thus, there exists a sequence (X_0, \dots, X_k) such that $X_0 = \emptyset$, $X_k = X$, and, for each $i \in [1, k]$, either X_i is an elementary collapse of X_{i-1} or X_{i-1} is an elementary collapse of X_i . Let $C \in \mathbb{C}$ such that we have $X_i \preceq C$ for each $i \in [0, k]$. The complex X_0 is a dendrite, suppose X_{i-1} is a dendrite for some $i \geq 1$.

i) If X_{i-1} is an elementary collapse of X_i , then $X_i = X_{i-1} \cup AB$, with $A \in \mathbb{C}[0]$, $AB \in \mathbb{C}$, $X_{i-1} \cap AB = AB^\circ$. By Prop. [6](#) AB and AB° are both dendrites. Thus, by $\langle C_{UP} \rangle$, X_i is a dendrite.

ii) If X_i is an elementary collapse of X_{i-1} , then, by Prop. [3](#) $(X_{i-1}; C)^*$ is an elementary collapse of $(X_i; C)^*$. Furthermore, by Prop. [11](#) $(X_{i-1}; C)^*$ is a dendrite. By using the arguments of i) in the dual, we may affirm that $(X_i; C)^*$ is a dendrite. By using again Prop. [11](#) it follows that X_i is a dendrite. \square

Theorem 2. *We have $\mathbb{D} = \langle \emptyset, \text{COL}, -\text{COL} \rangle$. In other words, a simplicial complex is a dendrite if and only if it is simply contractible.*

We define the one-to-one completion $\langle \text{DEF} \rangle$ which allows to make homotopic deformations of arbitrary complexes:

\rightarrow If $S \in \mathcal{K}$, then $S \cup D \in \mathcal{K}$ whenever $D \in \mathbb{D}$ and $S \cap D \in \mathbb{D}$. $\langle \text{DEF} \rangle$

Theorem 3. *Let $X, Y \in \mathbb{S}$. The complex Y is simple homotopic to X if and only if $Y \in \langle \{X\}, \text{DEF}, -\text{DEF} \rangle$.*

10 Conclusion

We have seen that (one-to-one) completions allow to formulate recursive transformations of objects. More remarkably, completions allow to define structures on objects (*e.g.*, a connection). We introduced two completions, $\langle C_{UP} \rangle$ and $\langle C_{AP} \rangle$, in order to define, in an axiomatic way, a collection of acyclic complexes. We gave a theorem which shows the equivalence between this collection and the collection made of simply contractible simplicial complexes, *i.e.*, complexes which may be transformed to a single point by collapse/anti-collapse operations. Thus, these two axioms may be used for expressing homotopic transforms such as the one illustrated Fig. [11](#). As we have seen in the introduction, this transformation cannot be interpreted as a continuous one when we consider the preorder associated to the classical axioms of topology.

References

1. Aczel, P.: An introduction to inductive definitions. In: Barwise, J. (ed.) Handbook of Mathematical Logic, pp. 739–782 (1977)
2. Alexandroff, P.: Diskrete Räume. Mat. Sbornik 2, 501–518 (1937)
3. Khalimsky, E.D.: On topologies of generalized segments. Soviet Math. Doklady 10, 1508–1511 (1969)

4. Kovalevsky, V.: Finite topology as applied to image analysis. In: Proc. of Comp. Vision Graphics, and Im., vol. 46, pp. 141–161 (1989)
5. Rosenfeld, A.: Digital topology. Amer. Math. Monthly, 621–630 (1979)
6. Tarski, A.: Logic, semantics and metamathematics. Oxford University Press, Oxford (1956)
7. Jonsson, J.: Simplicial Complexes of Graphs. Springer, Heidelberg (2008)
8. Björner, A.: Topological methods. In: Graham, R., Grötschel, M., Lovász, L. (eds.) Handbook of Combinatorics, pp. 1819–1872. North-Holland, Amsterdam (1995)
9. Hachimori, M.: Combinatorics of constructible complexes, PhD Thesis, Tokyo University (2000)
10. Hochster, M.: Rings of invariant of tori, Cohen-Macaulay rings generated by monomials, and polytopes. Ann. Math. 96, 318–337 (1972)
11. Hachimori, M.: Nonconstructible simplicial balls and a way of testing constructibility. Discrete Comp. Geom. 22, 223–230 (1999)
12. Whitehead, J.H.C.: Simplicial spaces, nuclei, and m -groups. Proc. London Math. Soc. 45(2), 243–327 (1939)
13. Bing, R.H.: Some aspects of the topology of 3-manifolds related to the Poincaré Conjecture. In: Saastoy, T.L. (ed.) Lectures on Modern Mathematics II, pp. 93–128. Wiley, Chichester (1964)
14. Zeeman, E.C.: On the dunce hat. Topology 2, 341–358 (1964)
15. Serra, J.: Image Analysis and Mathematical Morphology, part II: theoretical advances. Academic Press, London (1988)
16. Bertrand, G.: On critical kernels, Comptes Rendus de l'Académie des Sciences. Série Math. I(345), 363–367 (2007)
17. Couprie, M., Bertrand, G.: New characterizations of simple points in 2D, 3D and 4D discrete spaces. IEEE Transactions on PAMI 31(4), 637–648 (2009)
18. Kong, T.Y., Rosenfeld, A.: Digital topology: introduction and survey. In: Proc. of Comp. Vision, Graphics and Image, vol. 48, pp. 357–393 (1989)
19. Welker, V.: Constructions preserving evasiveness and collapsibility. Discrete Math. 207, 243–255 (1999)
20. Kahn, J., Saks, M., Sturtevant, D.: A topological approach to evasiveness. Combinatorica 4, 297–306 (1984)
21. Knaster, B.: Un théorème sur les fonctions d'ensembles. Ann. Soc. Pol. Math. 6, 133–134 (1928)
22. Tarski, A.: A lattice theoretical fixed point theorem and its applications. Pacific J. Math. 5, 285–309 (1955)
23. Kleene, S.C.: Introduction to Meta Mathematics. Van Nostrand, New-York (1962)
24. Lassez, J.-L., Nguyen, V.L., Sonenberg, E.A.: Fixed point theorems and semantics: a folk tale. Information Proc. Letters 14, 3 (1982)

Appendix A: Completions and Closure Operators

In the sequel, the symbol \mathbf{S} denotes an arbitrary collection.

Let γ be a map from $2^{\mathbf{S}}$ to $2^{\mathbf{S}}$, such a map is said to be an *operator (on \mathbf{S})*.

If γ is an operator on \mathbf{S} , we set $\gamma^1 = \gamma$ and $\gamma^k = \gamma \circ \gamma^{k-1}$, $k \geq 2$. We define $\widehat{\gamma}$ to be the operator such that, for each $\mathbf{X} \subseteq \mathbf{S}$, $\widehat{\gamma}(\mathbf{X}) = \cup\{\gamma^k(\mathbf{X}) \mid k \geq 1\}$.

Let γ be an operator on \mathbf{S} . We say that:

- γ is *extensive* if, for all $\mathbf{X} \subseteq \mathbf{S}$, we have $\mathbf{X} \subseteq \gamma(\mathbf{X})$.

- γ is *increasing* if, whenever $\mathbf{X} \subseteq \mathbf{Y} \subseteq \mathbf{S}$, we have $\gamma(\mathbf{X}) \subseteq \gamma(\mathbf{Y})$.
- γ is *idempotent* if, for all $\mathbf{X} \subseteq \mathbf{S}$, we have $\gamma^2(\mathbf{X}) = \gamma(\mathbf{X})$.

The operator γ is a *closure operator* (on \mathbf{S}) if γ is extensive, increasing, and idempotent. If γ is a closure operator, $\mathbf{X} \subseteq \mathbf{S}$ is *closed* for γ if $\gamma(\mathbf{X}) = \mathbf{X}$.

Let γ be a closure operator on \mathbf{S} . We have the following result, this is a basic property of closure operators:

For any $\mathbf{X} \subseteq \mathbf{S}$, we have $\gamma(\mathbf{X}) = \cap\{\mathbf{Y} \subseteq \mathbf{S} \mid \mathbf{X} \subseteq \mathbf{Y} \text{ and } \gamma(\mathbf{Y}) = \mathbf{Y}\}$.

Let γ be an operator on \mathbf{S} . We say that γ is *finitary* if, for all $\mathbf{X} \subseteq \mathbf{S}$, we have $\gamma(\mathbf{X}) = \cup\{\gamma(\mathbf{F}) \mid \mathbf{F} \subseteq \mathbf{X} \text{ and } \mathbf{F} \text{ finite}\}$.

Alfred Tarski [6] introduced finitary closure operators (also called “finite consequence operators”) as an abstract theory of logical deductions. In this context, the set \mathbf{S} represents a set of statements in some language. Given a subset \mathbf{X} of \mathbf{S} , the set $\gamma(\mathbf{X})$ represents the set of all statements that may be deduced from \mathbf{X} .

Observe that any finitary operator is increasing. Thus, an operator γ is a finitary closure operator if and only if γ is extensive, finitary, and idempotent.

We say that an operator κ is a Λ -operator if κ is extensive and finitary.

The following fixed point theorem is essential for our purpose. See [21-24] for more general fixed point theorems.

Theorem 4. *Let κ be a Λ -operator. Then, for any $\mathbf{X} \subseteq \mathbf{S}$, $\widehat{\kappa}(\mathbf{X})$ is a fixed point for κ , i.e., we have $\kappa(\widehat{\kappa}(\mathbf{X})) = \widehat{\kappa}(\mathbf{X})$.*

Proposition 12. *If κ is a Λ -operator, then $\widehat{\kappa}$ is a finitary closure operator.*

Remark 2. If γ is an extensive and increasing operator (not necessarily finitary), then $\widehat{\kappa}(\mathbf{X})$ is not necessarily a fixed point for κ .

Let κ be a binary relation over $2^{\mathbf{S}}$ and $2^{\mathbf{S}}$, thus $\kappa \subseteq 2^{\mathbf{S}} \times 2^{\mathbf{S}}$. We say that κ is a *constructor* (on \mathbf{S}) if κ is *finitary*, which means that \mathbf{F} is finite whenever $(\mathbf{F}, \mathbf{G}) \in \kappa$.

Let κ be a constructor on \mathbf{S} . We also denote by κ the Λ -operator such that, for each $\mathbf{X} \subseteq \mathbf{S}$, we have:

$$\kappa(\mathbf{X}) = \cup\{\mathbf{G} \mid \text{there exists } (\mathbf{F}, \mathbf{G}) \in \kappa \text{ with } \mathbf{F} \subseteq \mathbf{X}\} \cup \mathbf{X}.$$

We say that the operator κ is the Λ -operator induced by the constructor κ .

Let κ be a Λ -operator on \mathbf{S} , the *constructor induced by κ* is the constructor $\underline{\kappa}$ such that $\underline{\kappa} = \{(\mathbf{F}, \kappa(\mathbf{F})) \mid \mathbf{F} \text{ is a finite subset of } \mathbf{S}\}$.

The following result is a direct consequence of the above definitions. It shows that specifying a Λ -operator is (in a certain sense) equivalent to specifying a constructor.

Proposition 13. *Let κ be a Λ -operator, and let $\underline{\kappa}$ be the constructor induced by κ . The two Λ -operators $\underline{\kappa}$ and κ are equal.*

Let κ be a constructor on \mathbf{S} , and let $\underline{\kappa}$ be the constructor induced by the Λ -operator κ . The two Λ -operators κ and $\underline{\kappa}$ are equal.

Let κ be a constructor. We say that κ is *many-to-one* if $\text{Card}(\mathbf{G}) = 1$ whenever $(\mathbf{F}, \mathbf{G}) \in \kappa$.

To each arbitrary constructor \mathbf{k} we may associate a many-to-one constructor $\underline{\mathbf{k}}$, we define $\underline{\mathbf{k}}$ to be $\{(\mathbf{F}, \{\mathbf{x}\}) \mid \text{there exists } (\mathbf{F}, \mathbf{G}) \in \mathbf{k} \text{ and } \mathbf{x} \in \mathbf{G}\}$. We see that the two \mathcal{A} -operators \mathbf{k} and $\underline{\mathbf{k}}$ are equal. Thus, specifying a constructor is equivalent to specifying a many-to-one constructor.

Remark 3. Let \mathbf{k} be a binary relation over $2^{\mathbf{S}}$ and \mathbf{S} , thus $\mathbf{k} \subseteq 2^{\mathbf{S}} \times \mathbf{S}$. We say that \mathbf{k} is *finitary* if \mathbf{F} is finite whenever $(\mathbf{F}, \mathbf{x}) \in \mathbf{k}$. It may be seen that specifying a many-to-one constructor is equivalent to specifying such a relation.

In the sequel, the symbol \mathcal{K} will denote an arbitrary subcollection of \mathbf{S} , thus we have $\mathcal{K} \subseteq \mathbf{S}$.

If \mathbf{k} is a constructor on \mathbf{S} , we denote by $\langle \mathbf{k} \rangle$ the following property which is *the completion induced by \mathbf{k}* :

$$\rightarrow \text{If } \mathbf{F} \subseteq \mathcal{K}, \text{ then } \mathbf{G} \subseteq \mathcal{K} \text{ whenever } (\mathbf{F}, \mathbf{G}) \in \mathbf{k}. \quad \langle \mathbf{k} \rangle$$

Let $\langle \mathbf{K} \rangle$ be a property which depends on \mathcal{K} . We say that a given collection $\mathbf{X} \subseteq \mathbf{S}$ satisfies $\langle \mathbf{K} \rangle$ if the property $\langle \mathbf{K} \rangle$ is true for $\mathcal{K} = \mathbf{X}$.

Thus, if \mathbf{k} is a constructor, a collection $\mathbf{X} \subseteq \mathbf{S}$ satisfies $\langle \mathbf{k} \rangle$ if and only if $\mathbf{k}(\mathbf{X}) = \mathbf{X}$.

The following propositions are direct consequences of Th. 4 and Prop. 12. Prop. 14 and 15 are equivalent to Th. 11 given in section 2. The collection $\langle \mathbf{X}, \mathbf{K} \rangle$ of Th. 11 is equal to $\widehat{\mathbf{k}}(\mathbf{X})$.

Proposition 14. *Let \mathbf{k} be a constructor on \mathbf{S} and let $\mathbf{X} \subseteq \mathbf{S}$. Then $\widehat{\mathbf{k}}(\mathbf{X})$ is, under the subset ordering, the unique minimal collection which contains \mathbf{X} and which satisfies $\langle \mathbf{k} \rangle$.*

Let $\langle \mathbf{K} \rangle$ be a property which depends on \mathcal{K} .

If $\mathbf{X} \subseteq \mathbf{S}$, we set $\Delta(\mathbf{X}, \mathbf{K}) = \cap\{\mathbf{Y} \subseteq \mathbf{S} \mid \mathbf{X} \subseteq \mathbf{Y} \text{ and } \mathbf{Y} \text{ satisfies } \langle \mathbf{K} \rangle\}$.

Proposition 15. *If \mathbf{k} is a constructor on \mathbf{S} and if $\mathbf{X} \subseteq \mathbf{S}$, then $\widehat{\mathbf{k}}(\mathbf{X}) = \Delta(\mathbf{X}, \mathbf{K})$.*

We say that a property $\langle \mathbf{K} \rangle$ is a *completion (property)* if there exists a constructor \mathbf{k} such that $\langle \mathbf{K} \rangle = \langle \mathbf{k} \rangle$ which means that, for each $\mathcal{K} \subseteq \mathbf{S}$, $\langle \mathbf{K} \rangle$ is true if and only if $\langle \mathbf{k} \rangle$ is true.

Proposition 16. *Let $\langle \mathbf{K} \rangle$ be a property which depends on \mathcal{K} . The property $\langle \mathbf{K} \rangle$ is a completion if and only if, for each $\mathbf{X} \subseteq \mathbf{S}$:*

- i) $\Delta(\mathbf{X}, \mathbf{K})$ satisfies $\langle \mathbf{K} \rangle$, and*
- ii) $\Delta(\mathbf{X}, \mathbf{K}) = \cup\{\Delta(\mathbf{F}, \mathbf{K}) \mid \mathbf{F} \subseteq \mathbf{X} \text{ and } \mathbf{F} \text{ finite}\}$.*

Hierarchical Euclidean Skeletons in Cubical Complexes^{*}

Michel Couprie

Université Paris-Est, LIGM, Équipe A3SI, ESIEE Paris, France
m.couprie@esiee.fr

Abstract. In the 90s, several authors introduced the notion of a hierarchical family of 2D Euclidean skeletons, evolving smoothly under the control of a filtering parameter. We provide in this article a discrete framework which formalizes and generalizes this notion, in particular to higher dimensions. This framework allows us to propose a new skeletonization scheme and to prove several important properties, such as topology preservation and stability w.r.t. parameter changes.

Despite the simplicity of the most common definition of a skeleton, as the set of all centers of maximal included balls, its use in real applications often raises difficult problems. One of the main issues is its lack of stability.

In 2005, F. Chazal and A. Lieutier introduced the λ -medial axis [4], a particular class of filtered skeletons, and studied its properties, in particular those related to stability. A major outcome of [4] is the following property: informally, except for particular values of the filtering parameter, the λ -medial axis remains stable under perturbations of the shape that are small with regard to the Hausdorff distance. However, the original definition of the λ -medial axis only holds and makes sense in the (continuous) Euclidean n -dimensional space. In [3], J. Chaussard *et al.* introduced the definition of a discrete λ -medial axis in \mathbb{Z}^n , and showed that it provides good robustness to boundary noise and invariance by rotation in practice.

For certain applications however, the λ -medial axis cannot be used because, when a small modification of the filtering parameter λ is done around some particular values, topological changes or abrupt appearing/disappearing of branches typically occur in the resulting skeleton.

On the other hand, several authors have proposed a general idea that leads to the notion of a family of skeletons, evolving smoothly under the control of a single parameter. They called it hierarchic skeletons [10], veinerization [11] or multiscale skeletons [8]. Such a family of skeletons can be summarized by a function, called potential residual in [10]. The skeletons are obtained as level sets (*i.e.*, thresholds) of this function. The method of R.L. Ogniewicz and O. Kübler [10] applies to 2D shapes that are (sets of) planar polygons in \mathbb{R}^2 , and the resulting skeleton is a set of straight line segments which do not necessarily

^{*} This work has been partially supported by the “ANR BLAN07-2_184378 MicroFiss” project.

have their extremities on a grid. The authors give a proof that the obtained skeleton is homotopy-equivalent to the original shape, but discretizations of these skeletons in \mathbb{Z}^2 do not generally share the same property. Veinerization [11] and multiscale skeletons [8] are methods that operate in the 2D square grid, and that are based on the same general idea as Ogniewicz and Kübler's. However [11] does not provide an algorithm to compute skeletons in practice, and the algorithm proposed in [8] does not guarantee topology preservation. Extensions of these works to higher-dimensional spaces (*e.g.*, 3D) have not been considered so far, to our knowledge.

In this article, we propose an effective and flexible method for building and filtering discrete skeletons, in the framework of cubical complexes. We ensure topology preservation by founding our method on the collapse operation, an elementary deformation that preserves the homotopy type.

First, we propose a thinning procedure that combines three ideas: λ -medial axis (Sec. 2), directional parallel thinning, and guided thinning (Sec. 3). This procedure also produces a sequence of collapse operations from which we derive an acyclic graph structure that we call a flow graph.

Based on this flow graph, we define the notion of topological map. We prove that if M is a topological map on an object X , then any level set of M has the same homotopy type as X (Th. 5). Moreover, we prove that if the real numbers a and b are close to each other, then the shapes M_a and M_b (the level sets of M at levels a and b) are close to each other¹ with respect to the Hausdorff distance (Th. 6). This property will assess the stability of our final skeletonization method w.r.t. the parameter value.

Then, we show how to build particular topological maps based on different measures of shape characteristics, thanks to the notion of upstream (Sec. 3, Sec. 6).

Finally, we propose a skeletonization scheme that produces families of filtered homotopic skeletons (Sec. 7). A filtered skeleton is obtained as a level set of the obtained topological map.

Proofs of the stated properties, experimental evaluation, and more illustrations can be found in an extended version of this article [6].

1 Cubical Complexes and Collapse

In this section, we recall briefly some basic definitions on cubical complexes, see also [1] for more details.

Let S be a set. If T is a subset of S , we write $T \subseteq S$. We denote by $|S|$ the number of elements of S .

Let \mathbb{Z} be the set of integers. We consider the families of sets $\mathbb{F}_0^1, \mathbb{F}_1^1$, such that $\mathbb{F}_0^1 = \{\{a\} \mid a \in \mathbb{Z}\}$, $\mathbb{F}_1^1 = \{\{a, a+1\} \mid a \in \mathbb{Z}\}$. A subset f of \mathbb{Z}^n , $n \geq 2$, which is the Cartesian product of exactly m elements of \mathbb{F}_1^1 and $(n-m)$ elements of \mathbb{F}_0^1 is called a *face* or an *m-face* in \mathbb{Z}^n , m is the *dimension* of f , we write $\dim(f) = m$.

¹ In the sense of Lipschitz continuity.

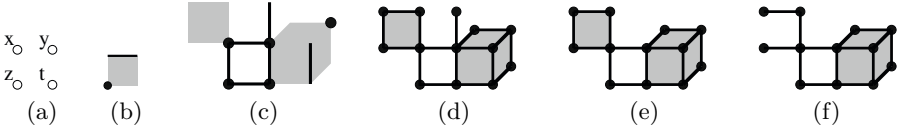


Fig. 1. (a) Four points in \mathbb{Z}^2 : $x = (0, 1)$; $y = (1, 1)$; $z = (0, 0)$; $t = (1, 0)$. (b) A graphical representation of the set of faces $\{f_0, f_1, f_2\}$, where $f_0 = \{z\} = \{0\} \times \{0\}$ (a 0-face), $f_1 = \{x, y\} = \{0, 1\} \times \{1\}$ (a 1-face), and $f_2 = \{x, y, z, t\} = \{0, 1\} \times \{0, 1\}$ (a 2-face). (c) A set of faces that is not a complex. (d) A set of faces that is a complex. (e, f) Two steps of elementary collapse from (d).

We denote by \mathbb{F}^n the set composed of all faces in \mathbb{Z}^n . An m -face is called a *point* if $m = 0$, a (*unit*) *edge* if $m = 1$, a (*unit*) *square* if $m = 2$, a (*unit*) *cube* if $m = 3$.

Let f be a face in \mathbb{F}^n . We set $\hat{f} = \{g \in \mathbb{F}^n \mid g \subseteq f\}$ and $\hat{f}^* = \hat{f} \setminus \{f\}$. Any $g \in \hat{f}$ is called a *face of f*.

A finite set X of faces in \mathbb{F}^n is a *complex (in \mathbb{F}^n)* if for each face $f \in X$, we have $\hat{f} \subseteq X$. See in Fig. 1(d,e,f) some examples of complexes, and in Fig. 1(b,c) examples of sets of faces that are not complexes.

The *collapse operation* is an elementary topology-preserving transformation which has been introduced by J.H.C. Whitehead [14] and plays an important role in combinatorial topology. It can be seen as a discrete analogue of a continuous deformation (a strong deformation retract). Collapse is known to preserve the homotopy type.

Let X be a complex in \mathbb{F}^n and let $(f, g) \in X^2$. If f is the only face of X that strictly includes g , then g is said to be *free for X* and the pair (f, g) is said to be a *free pair for X*. Notice that, if (f, g) is a free pair, then we have necessarily $\dim(g) = \dim(f) - 1$.

Let X be a complex, and let (f, g) be a free pair for X . Let $m = \dim(f)$. The complex $X \setminus \{f, g\}$ is an *elementary collapse of X*, or an *elementary m-collapse of X* (see Fig. 1(e) is an elementary 1-collapse of (d), (f) is an elementary 2-collapse of (e)).

Let X, Y be two complexes. We say that X *collapses onto Y*, and we write $X \searrow Y$, if $Y = X$ or if there exists a *collapse sequence from X to Y*, i.e., a sequence of complexes $\langle X_0, \dots, X_\ell \rangle$ such that $X_0 = X$, $X_\ell = Y$, and X_i is an elementary collapse of X_{i-1} , for each $i \in \{1, \dots, \ell\}$. Let $J = \langle (f_i, g_i) \rangle_{i=1}^\ell$ be the sequence of pairs of faces of X such that $X_i = X_{i-1} \setminus \{f_i, g_i\}$, for any $i \in \{1, \dots, \ell\}$. We also call the sequence J a *collapse sequence (from X to Y)*.

2 The Discrete λ -Medial Axis and the Projection Radius Map

The original definition of the λ -medial axis (see [4]) holds and makes sense in the (continuous) Euclidean n -dimensional space. The definition of a discrete λ -medial axis (DLMA) in \mathbb{Z}^n is given in [3], together with an experimental evaluation of its stability and rotation invariance.

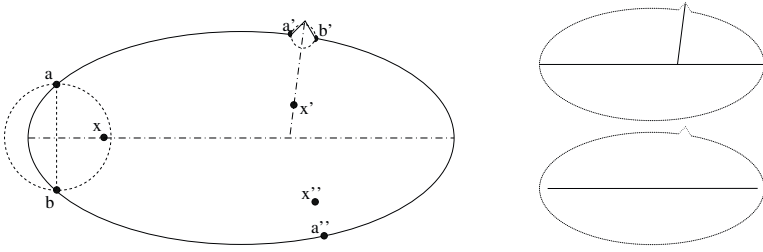


Fig. 2. Illustration of the λ -medial axis. Left: Points x, x' and x'' and their respective closest boundary points. Top right: λ -medial axis with $\lambda = \epsilon$, a very small positive real number. Bottom right: λ -medial axis with $\lambda = d(a', b') + \epsilon$.

Notice that the DLMA applies on a digital image (*i.e.*, a subset of \mathbb{Z}^n), not on a complex. However, the bijective correspondance between elements of \mathbb{Z}^n and n -faces in \mathbb{F}^n allows us to use the DLMA and related notions in the context of cubical complexes.

Let $x = (x_1, \dots, x_n), y = (y_1, \dots, y_n) \in \mathbb{R}^n$, we denote by $d(x, y)$ the Euclidean distance between x and y , in other words, $d(x, y) = (\sum_{k=1}^n (y_k - x_k)^2)^{\frac{1}{2}}$. Let $S \subseteq \mathbb{R}^n$, we set $d(y, S) = \min_{x \in S} \{d(y, x)\}$.

Let $x \in \mathbb{R}^n, r \in \mathbb{R}^+$, we denote by $B_r(x)$ the ball of radius r centered on x , defined by $B_r(x) = \{y \in \mathbb{R}^n \mid d(x, y) \leq r\}$.

Let S be a nonempty subset of \mathbb{R}^n , and let $x \in \mathbb{R}^n$. The projection of x on S , denoted by $\Pi_S(x)$, is the set of points y of S which are at minimal distance from x ; more precisely,

$$\Pi_S(x) = \{y \in S \mid \forall z \in S, d(y, x) \leq d(z, x)\}.$$

Let X be an open bounded subset of \mathbb{R}^n , and let $\lambda \in \mathbb{R}^+$. We denote by \overline{X} the complement set of X , *i.e.*, $\overline{X} = \mathbb{R}^n \setminus X$. The λ -medial axis of X is the set of points x in X such that the radius of the smallest ball that includes $\Pi_{\overline{X}}(x)$ is not less than λ (see Fig. 2).

For each point $x \in \mathbb{Z}^n$, we define the direct neighborhood of x as $N(x) = \{y \in \mathbb{Z}^n \mid d(x, y) \leq 1\}$.

Transposing directly the definition of the λ -medial axis to the discrete grid \mathbb{Z}^n would yield unsatisfactory results (see [3]), this is why we need the following notion. Let $S \subseteq \mathbb{Z}^n$, and let $x \in S$. The extended projection of x on \overline{S} (where $\overline{S} = \mathbb{Z}^n \setminus S$), denoted by $\Pi_{\overline{S}}^e(x)$, is the union of the sets $\Pi_{\overline{S}}(y)$, for all y in $N(x)$ such that $d(y, \overline{S}) \leq d(x, \overline{S})$.

Let S be a finite subset of \mathbb{Z}^n , and let $\lambda \in \mathbb{R}^+$. We define the function PR_S which associates, to each point x of S , the value $PR_S(x)$ that is the radius of the smallest ball enclosing all the points of the extended projection of x on \overline{S} . In other terms, $PR_S(x) = \min\{r \in \mathbb{R}^+ \mid \exists y \in \mathbb{R}^n, \Pi_{\overline{S}}^e(x) \subseteq B_r(y)\}$, and we call $PR_S(x)$ the projection radius of x (for S). The discrete λ -medial axis of S , denoted by $DLMA(S, \lambda)$, is the set of points x in S such that $PR_S(x) \geq \lambda$.

Note that the function PR_S can be computed once and stored as a grayscale image, and that any DLMA of S is a level set of this function at a particular

value λ . Notice also that DLMA has not, in general, the same topology as the original shape. For more details, illustrations and performance analysis, see [3].

3 Guided Collapse and Flow Graph

In this section, we introduce a thinning scheme that produces a collapse sequence, based on an arbitrary priority map (e.g., a distance map or a projection radius map). The general idea of guided thinning is not new: it has been used by several authors to produce skeletons based on the Euclidean distance [7][13][12], and consists of using the priority function in order to specify which elements must be considered at each step of the thinning. Here, we combine this general idea with a parallel directional collapse algorithm introduced in [2], in order to minimize the number of arbitrary decisions. When several elements share the same priority, which may occur quite often, we remove in parallel all such elements that satisfy a condition based on direction and dimension. All directions and dimensions are successively explored.

First, we need to define the direction of a free face. Let X be a complex in \mathbb{F}^n , let (f, g) be a free pair for X . Since (f, g) is free, we know that $\dim(g) = \dim(f) - 1$, and it can be easily seen that $f = g \cup g'$ where g' is the translate of g by one of the $2n$ vectors of \mathbb{Z}^n with all coordinates equal to 0 except one, which is either $+1$ or -1 . Let v denote this vector, and c its non-null coordinate. We define $Dir(f, g)$ as the index of c in v , it is the *direction* of the free pair (f, g) . Its *orientation* is defined as $Orient(f, g) = 1$ if $c = +1$, and as $Orient(f, g) = 0$ otherwise.

Now, we are ready to introduce algorithm *GuidedCollapse* (see Alg. [1]). The symbol $+$ is used to denote the action of appending an element at the end of a sequence.

We have the following property.

Proposition 1. *Whatever the complex X and the map P from X to \mathbb{R} , X collapses onto $GuidedCollapse(X, P)$.*

Algorithm *GuidedCollapse* may be implemented to run in $O(N \log N)$ time complexity, where N denotes the cardinality of X , using a balanced binary tree data structure (see [5]) for representing the set R .

To conclude this section, we introduce the notion of a flow graph associated to a given collapse sequence.

A (*finite directed*) *graph* is a pair (V, E) where V is a finite set and E is a subset of $V \times V$. An element of V is called a *vertex*, an element of E is called an *arc*. A *path* in (V, E) is a sequence $\langle v_i \rangle_{i=0}^\ell$ of vertices such that $\ell \geq 0$ and for all $i \in \{1, \dots, \ell\}$, we have $(v_{i-1}, v_i) \in E$. The number ℓ is the *length* of the path. If $\ell = 0$ the path is said *trivial*. If $v_0 = v_\ell$ the path is a *cycle*. The graph is *acyclic* if it does not contain any non-trivial cycle.

Definition 2. *Let X be a complex and $J = \langle (f_i, g_i) \rangle_{i=1}^\ell$ be a collapse sequence from X . For any $k \in \{1, \dots, \ell\}$, $\ell \geq 0$, we set $X_k = X \setminus \{f_i, g_i\}_{i=1}^k$. We set $E_1 = \{(g_i, f_i)\}_{i=1}^\ell$ and $E_2 = \bigcup_{k=1}^\ell \{(f_k, g) \mid g \in \hat{f}_k^* \cap X_k\}$.*

Algorithm 1. *GuidedCollapse*(X, P)

Data: A cubical complex X in \mathbb{F}^n , and a map P from X to \mathbb{R} (priority map)
Result: A collapse sequence J

```

1  $J = \langle \rangle$ ;  $R = \{(p, f, g) \mid (f, g) \text{ is free for } X, p = \max(P(f), P(g))\}$ ;
2 while  $R \neq \emptyset$  do
3    $m = \min\{p \mid (p, \dots) \in R\}$ ;  $Q = \{(m, \dots) \in R\}$ ;  $R = R \setminus Q$ ;
4    $L = \{(f, g) \mid (\dots, f, g) \in Q\}$ ;
5   for  $t = 1 \rightarrow n$  /* direction */ do
6     for  $s = 0 \rightarrow 1$  /* orientation */ do
7       for  $d = n \rightarrow 1$  /* decreasing dimension */ do
8          $T = \{(f, g) \in L \mid (f, g) \text{ is free for } X,$ 
9            $\text{Dir}(f, g) = t, \text{Orient}(f, g) = s, \dim(f) = d\}$ ;
10         $X = X \setminus T$ ;
11        foreach  $(f, g) \in T$  do  $J = J + (f, g)$ ;
12        foreach  $(i, j) \in X^2, j \in \hat{f}^*$  do
13          if  $(i, j)$  is free for  $X$  then
14             $p = \max(P(i), P(j))$ ;
15            if  $p \leq m$  then  $L = L \cup \{(i, j)\}$ ;
16             $R = R \cup \{(p, i, j)\}$ ;
16 return  $J$ ;

```

The flow graph associated to J is the (directed) graph whose vertex set is X and whose edge set is $E = E_1 \cup E_2$.

This definition is illustrated in Fig. 3. It can be easily seen that, whatever the complex X and the collapse sequence J from X , the flow graph associated to J is acyclic.

4 Upstream of a Vertex and Its Valuation

From now, we consider a collapse sequence $J = \langle (f_i, g_i) \rangle_{i=1}^\ell$ from a complex X , and its associated flow graph $(X, E = E_1 \cup E_2)$. Using the notations of Def. 2, any pair (f_k, g_k) of J is free for X_{k-1} , and we have $X = X_0 \searrow \dots \searrow X_\ell$. We define $F = \{f_i\}_{i=1}^\ell$, $G = \{g_i\}_{i=1}^\ell$ and $X_J = F \cup G$.

Let $x \in X$, we denote by $\Gamma(x)$ the set of *successors* of x in the acyclic graph (X, E) , that is, $\Gamma(x) = \{y \in X \mid (x, y) \in E\}$, and we denote by $\Gamma^{-1}(x)$ the set of *predecessors* of x in this graph, that is, $\Gamma^{-1}(x) = \{y \in X \mid (y, x) \in E\}$. We denote by $d^+(x)$ the *outer degree* of the vertex x in the graph (X, E) , that is, the number of successors of x .

We call *upstream* of x the set of all vertices that are ancestors of x in the flow graph, that is, the set $U(x) = \{y \in X \mid \text{there is a path from } y \text{ to } x \text{ in } (X, E)\}$.

In a collapse sequence, certain pairs can be swapped or eliminated, yielding another collapse sequence. Intuitively, the elements of the upstream of a face x of X are those that must indeed be collapsed before x can itself collapse.

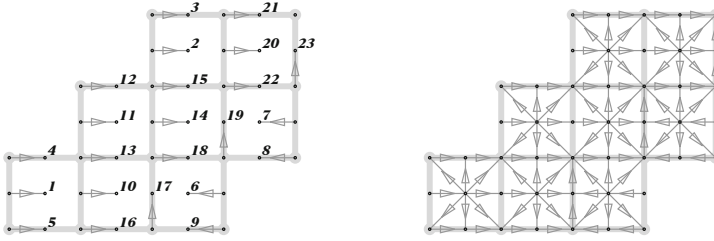


Fig. 3. Left: A collapse sequence J . Each pair (f_i, g_i) of J is depicted by an arrow from g_i to f_i . The numbers indicate the indices of the pairs in J . Right: The flow graph associated to J .

Let L be a map from X to $\mathbb{R} \cup \{+\infty\}$. Roughly speaking, the map \tilde{L} defined below cumulates, for each vertex x , the values of L on all vertices of the upstream of x .

Definition 3. Let L be a map from X to $\mathbb{R} \cup \{+\infty\}$. We define the map \tilde{L} such that, for any $x \in X$:

$$\tilde{L}(x) = L(x) + \sum_{y \in \Gamma^{-1}(x)} \tilde{L}(y)/d^+(y)$$

Notice that this definition is recursive, and that it makes sense since the graph (X, E) is acyclic. Intuitively, the division by $d^+(y)$ is motivated by the fact that a value must not be taken in account several times in the sum. The values $\tilde{L}(x)$ can be computed by a quite simple recursive program (given in [6]) that has a linear time complexity.

Two particularly simple functions L yield meaningful indicators associated to the elements of X . Let us first consider the function L_1 such that $L_1(x) = 1$ if $\dim(x) = n$, and $L_1(x) = 0$ otherwise. The map \tilde{L}_1 associates, to each element x of X , the “area of $U(x)$ ” (or its volume in 3D). Now, let us consider $L_2 = 1_{B(X)}$, where $B(X)$ is the set of all faces that are free for X . We call $B(X)$ the *border of X* . In other words, $L_2(x) = 1$ if $x \in B(X)$, and $L_2(x) = 0$ otherwise. The map \tilde{L}_2 associates, to each element x of X , a measure (length in 2D, surface area in 3D) of $U(x) \cap B(X)$.

Fig. 4(a₁, a₂) show the maps L_1 and L_2 respectively, for the same object Y . The maps \tilde{L}_1 and \tilde{L}_2 are displayed in Fig. 4(b₁, b₂).

5 Topological Maps

In this section, we introduce the notion of topological map. A topological map based on a collapse sequence J is a map on the elements of X that satisfies certain conditions relative to J and its associated flow graph. Then, we prove an important property of such maps: if M is a topological map, then any level set of M is homotopy-equivalent to X . In Sec. 6, we will show how to build such a map, based on any given function on X .

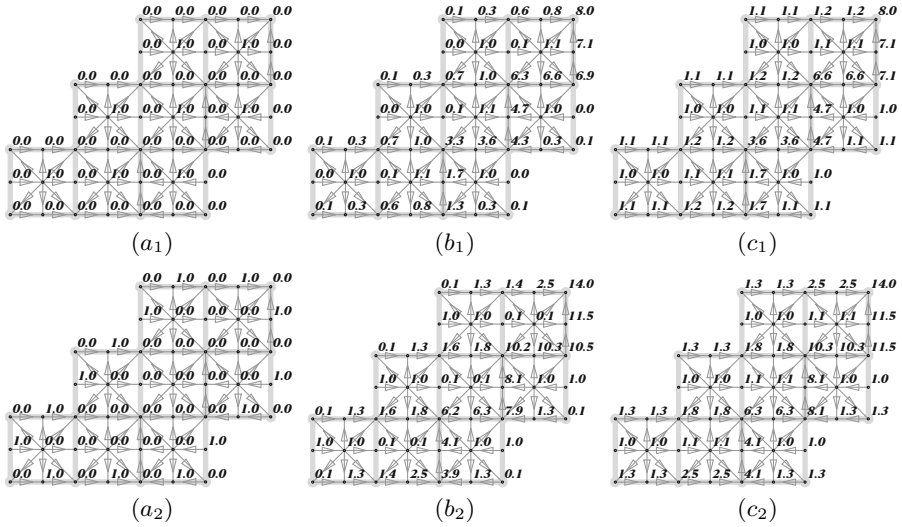


Fig. 4. (a_1, a_2) Maps L_1 and L_2 on the same complex Y . (b_1, b_2) Maps \tilde{L}_1 and \tilde{L}_2 . For the sake of readability, only one digit after the decimal point is displayed. (c_1, c_2) α -Topological maps induced by \tilde{L}_1 and \tilde{L}_2 , respectively, with $\alpha = 0.1$ (see Sec. 5 and Sec. 6).

Definition 4. Let M be a map from X to $\mathbb{R} \cup \{+\infty\}$. We say that M is a topological map on X (based on J) if:

- i)* for all (g, f) in E_1 , $M(g) = M(f)$; and
- ii)* for all (f, g) in E_2 , $M(g) > M(f)$; and
- iii)* for all g in $X \setminus X_J$, $M(g) = +\infty$.

Let α be a positive real number. If we replace *ii)* with the stronger requirement: *ii')* for all (f, g) in E_2 , $M(g) \geq M(f) + \alpha$, then we say that M is an α -topological map on X (based on J).

The notion of topological map is inspired from the one of discrete Morse function (see [9]). A topological map can be seen (apart from the infinite values) as a particular case of discrete Morse function, and Th. 5 could also be proved using results of [9].

Let $\lambda \in \mathbb{R} \cup \{+\infty\}$, we define $M_\lambda = \{x \in X \mid M(x) \geq \lambda\}$, the (*upper*) level set of M at level λ . The main property of a topological map M is that any level set of M is homotopy-equivalent to X , as implied by the following theorem (Th. 5, see Fig. 5 for an illustration).

Theorem 5. Let M be a topological map on X . Whatever the number $\lambda \in \mathbb{R} \cup \{+\infty\}$, the complex X collapses onto M_λ .

The next theorem (Th. 6) expresses the stability of our skeletonization scheme, with respect to the variations of the filtering parameter.

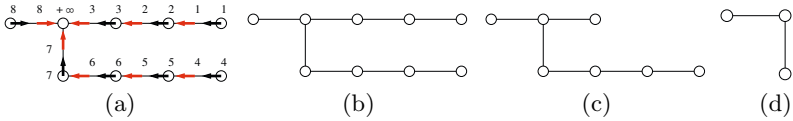


Fig. 5. (a) A 1-complex X , a flow graph on X (black arrows for arcs of E_1 , red arrows for arcs of E_2), and a (1)-topological map M on X (numbers). (b,c,d) Level sets of M at levels 0, 3 and 7, respectively.

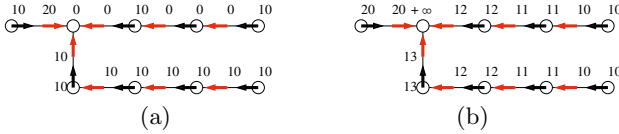


Fig. 6. (a) A map L on the complex X of Fig. 5. (b) The 1-topological map induced by L .

Let S, T be two subsets of \mathbb{R}^n . We set $H(S|T) = \max_{s \in S} \{\min_{t \in T} \{d(s, t)\}\}$, and $d_H(S, T) = \max\{H(S|T), H(T|S)\}$ is the *Hausdorff distance between S and T* . Let X be a complex in \mathbb{F}^n , we denote by $\mathcal{S}(X)$ the union of all faces of X , called the *support of X* . For comparing two complexes X and Y , we consider the Hausdorff distance between their supports $\mathcal{S}(X)$ and $\mathcal{S}(Y)$.

Theorem 6. *Let $\alpha \in \mathbb{R}, \lambda \in \mathbb{R} \cup \{+\infty\}, \alpha > 0, \lambda \geq 0$. Let $k \in \mathbb{N}$. Let M be an α -topological map on X . Then, $d_H(\mathcal{S}(M_\lambda), \mathcal{S}(M_{\lambda+k\alpha})) \leq k$.*

6 Topological Map Induced by an Arbitrary Map

In this section, we show that given any map L on X , we can define and compute a topological map that is “close to” L , more precisely it is the lowest map above L that is a topological map.

Definition 7. *Let L be any map from X to $\mathbb{R} \cup \{+\infty\}$, and let α be a positive real number. We consider a map M such that:*

- a) M is an α -topological map; and
- b) for all f in X_J , $M(f) \geq L(f)$; and
- c) M is minimal for conditions a) and b), that is, any map M' verifying both a) and b) is such that $M' \geq M$.

As stated by the following property, M is uniquely defined. We say that the map M is the α -topological map induced by L .

Proposition 8. *Let M and M' be two maps that verify conditions a), b) and c) of Def. 7. Then, we have $M = M'$.*

This notion is illustrated in Fig. 6 and in Fig. 4(c_1, c_2).

Next, we give an algorithm (Alg. 2) that computes the α -topological map induced by any given map on X . Before this, let us recall briefly the notions of

Algorithm 2. *AlphaTM*(X, E_1, E_2, L, α)

Data: A complex X , the arc sets E_1, E_2 of a flow graph on X , a map L from X to \mathbb{R} , a real number $\alpha > 0$

Result: A topological map M

- 1 **foreach** $x \in X$ **do**
 - ┌ **if** x does not appear in $E_1 \cup E_2$ **then** $M(x) = +\infty$;
 - └ **else** $M(x) = L(x)$;
- 2 Let $\{X^r\}_{r=0}^k$ be the result of the topological sort of the acyclic graph $(X, E_1 \cup E_2)$;
- 3 **for** $r = 0 \rightarrow k$ **do**
- 4 **foreach** $x \in X^r$ **do**
- 5 ┌ **foreach** y such that $(y, x) \in E_1$ **do**
- 6 └ ┌ $M(y) = M(x) = \max\{M(x), M(y)\}$;
- 6 └ ┌ **foreach** y such that $(y, x) \in E_2$ **do**
- 6 └ └ ┌ $M(x) = \max\{M(x), M(y) + \alpha\}$;
- 7 **return** M ;

Algorithm 3. *TopoMap*(X, L, α)

Data: A complex X , a map L on X , a real number α

Result: A topological map M

- 1 Let P be the projection radius map of X (see Sec. 2);
- 2 Let $J = \text{GuidedCollapse}(X, P)$ (see Sec. 3);
- 3 Let $(X, E = E_1 \cup E_2)$ be the flow graph associated to J ;
- 4 Let $M = \text{AlphaTM}(X, E_1, E_2, \tilde{L}, \alpha)$ (see Sec. 5);
- 5 **return** M ;

rank and topological sort (an introduction to topological sort, including definition, properties and algorithm, can be found, e.g., in [5]). Let $G = (V, E)$ be an acyclic graph and let $x \in V$, the *rank of x in G* is the length of the longest path in G that ends in x . The *topological sort of G* is an operation that results in a partition $\{V^r\}_{r=0}^k$ of V such that each V^r is the subset of V containing all vertices of rank r .

Proposition 9. *Let L be a map from X to \mathbb{R} , and let α be real number, $\alpha > 0$. The result of $\text{AlphaTM}(X, E_1, E_2, L, \alpha)$ is the α -topological map induced by L .*

7 Computing Hierarchic Skeletons

Let us now summarize our method to produce families of filtered homotopic skeletons (see algorithm 3). It is assumed here that X is a pure n -complex in \mathbb{F}^n , that is, a complex in which each face is included in an n -face.

First, we compute the projection radius map (Sec. 2) on the n -faces of X , and extend it to the other elements of X (if $y \in X$ is not an n -face, then we set $P(y)$ to the max of $P(x_i)$ where the x_i 's are all n -faces that include y).

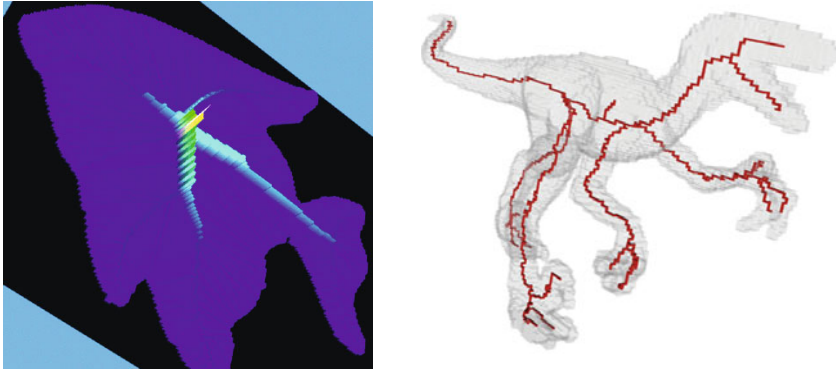


Fig. 7. Left: a rendering of the result of the $TopoMap$ operator, using the map L_2 . Right: a filtered curvilinear skeleton of a 3D shape.

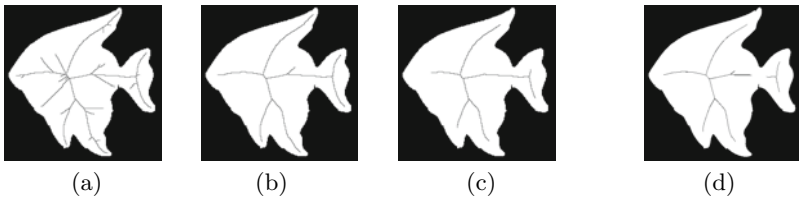


Fig. 8. (a,b,c) Three level sets of $TopoMap(X, L_2)$, at values 25, 50 and 130, respectively. (d) DLMA of X , with $\lambda = 45$.

Using algorithm [1](#) (Sec. [3](#)) we build a collapse sequence and a flow graph on X . By construction, the upstream of any vertex x of this flow graph is composed by elements that, in any family of filtered skeletons, should disappear before x does.

Integrating information given by map L allows us to associate, to each element x of X , a value $\tilde{L}(x)$ that represents a measure of the upstream of x . The lower this value, the sooner the point x may disappear.

Then, thanks to algorithm [2](#) (Sec. [5](#)), we produce a topological map M based on this measure. Due to Th. [5](#), we know that any level set of M is homotopy-equivalent to X . Therefore, filtered (*i.e.*, pruned) skeletons are obtained by thresholding the map M ; lowest levels of threshold correspond to highest levels of detail. Some results are shown in Fig. [7](#) and Fig. [8](#).

8 Conclusion

The method that we propose is guaranteed to preserve topology and is stable with respect to variations of the filtering parameter, as stated by Th. [5](#) and Th. [6](#) respectively. It is designed to work in 3D as well as in 2D. Furthermore, our method is highly flexible: many variants can be imagined, in particular by

choosing alternative valuations of the upstream. Future work include comparison with other methods for computing discrete filtered Euclidean skeletons.

References

1. Bertrand, G., Couprie, M.: A new 3D parallel thinning scheme based on critical kernels. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) DGC 2006. LNCS, vol. 4245, pp. 580–591. Springer, Heidelberg (2006)
2. Chaussard, J., Couprie, M.: Surface thinning in 3D cubical complexes. In: Wiederhold, P., Barneva, R.P. (eds.) IWCIA 2009. LNCS, vol. 5852, pp. 135–148. Springer, Heidelberg (2009)
3. Chaussard, J., Couprie, M., Talbot, H.: Robust skeletonization using the discrete lambda-medial axis. *Pattern Recognition Letters*, 1–10 (2010) (to appear), doi.org/10.1016/j.patrec.2010.09.002
4. Chazal, F., Lieutier, A.: The lambda medial axis. *Graphical Models* 67(4), 304–331 (2005)
5. Cormen, T.H., Leiserson, C., Rivest, R.: *Introduction to algorithms*. MIT Press, Cambridge (1990)
6. Couprie, M.: Topological maps and robust Euclidean skeletons in cubical complexes (2010), <http://www.esiee.fr/~couprie/Pdf/topomaps.pdf>
7. Davies, E.R., Plummer, A.P.N.: Thinning algorithms: a critique and a new methodology. *Pattern Recognition* 14, 53–63 (1981)
8. Falcao, A.X., da Fontoura Costa, L., da Cunha, B.S.: Multiscale skeletons by image foresting transform and its application to neuromorphometry. *Pattern Recognition* 35(7), 1571–1582 (2002)
9. Forman, R.: Morse theory for cell complexes. *Advances in Mathematics* 134(1), 90–145 (1998)
10. Ogniewicz, R.L., Kübler, O.: Hierarchic Voronoi skeletons. *Pattern Recognition* 28(33), 343–359 (1995)
11. Pierrot-Deseilligny, M., Stamon, G., Suen, C.Y.: Veinerization: a new shape description for flexible skeletonization. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 20(5), 505–521 (1998)
12. Pudney, C.: Distance-ordered homotopic thinning: a skeletonization algorithm for 3D digital images. *Comp. Vision and Image Understanding* 72(3), 404–413 (1998)
13. Talbot, H., Vincent, L.: Euclidean skeletons and conditional bisectors. In: *Procs. VCIP 1992, SPIE*, vol. 1818, pp. 862–876 (1992)
14. Whitehead, J.H.C.: Simplicial spaces, nuclei and m -groups. *Procs. of the London Mathematical Society* 45(2), 243–327 (1939)

Well-Composed Cell Complexes

Rocio Gonzalez-Diaz, Maria-Jose Jimenez, and Belen Medrano

Applied Math Department, University of Seville,
Av. Reina Mercedes, s/n, Seville, Spain
{rogodi,majiro,belenmg}@us.es

Abstract. Well-composed 3D digital images, which are 3D binary digital images whose boundary surface is made up by 2D manifolds, enjoy important topological and geometric properties that turn out to be advantageous for some applications. In this paper, we present a method to transform the cubical complex associated to a 3D binary digital image (which is not generally a well-composed image) into a cell complex that is homotopy equivalent to the first one and whose boundary surface is composed by 2D manifolds. This way, the new representation of the digital image can benefit from the application of algorithms that are developed over surfaces embedded in \mathbb{R}^3 .

Keywords: Well-composed digital images, cubical complex, cell complex, homotopy equivalence.

1 Introduction

We are mainly interested in studying topological features of 3D digital images. More concretely, our ultimate purpose is that of extracting cohomological information of a 3D model that could be used in characterization or recognition tasks (see [5,6,4,7] as related works). For this aim, it would be useful to compute first geometrically relevant representative cycles of homology generators of dimension 1 in the surface of the model, since this could simplify cohomological computations. Many applications such as topology repair, surface parameterization and feature recognition benefit from computing loops on surfaces that wrap around their ‘handles’ and ‘tunnels’ (defined by Dey et al. in [2]). In the paper [3], there is an algorithm to compute topologically correct loops that are also geometrically relevant in that sense. A refinement of the algorithm is given in [1]. However, all the computations are carried out over a connected closed surface in \mathbb{R}^3 . Since we are interested in applying these results to 3D binary digital images, we focus in the so-called *well-composed* images. A 3D binary digital image is said to be well-composed if and only if the square faces shared by foreground and background voxels of the image form a 2D manifold. Well-composed images enjoy important topological and geometric properties: there is only one type of connected component in any well-composed image, and hence, several algorithms used in computer vision, computer graphics and image processing are simpler; thinning algorithms can be simplified and naturally made parallel if the input

image is well-composed [10,13]; some algorithms for computing surface curvature or extracting adaptive triangulated surfaces [8], assume that the input image is well-composed. Since 2D and 3D images are often not well-composed images, there are several methods (repairing algorithms) for turning 3D binary digital images that are not well-composed into well-composed ones (see [12,15]), but these methods do not guarantee the topological equivalence between the original object and its corresponding well-composed image. In fact, the purpose can even be to simplify as much as possible the topology in the sense of removing little topological artifacts from the image. However we are concerned with the fact of preserving the topology of the input image having in mind cases in which subtle details can be important.

2 3D Digital Images and Cubical Complexes

Consider \mathbb{Z}^3 as the set of points with integer coordinates in 3D space \mathbb{R}^3 . A 3D binary digital image $I = (\mathbb{Z}^3, 26, 6, B)$ (or $I = (\mathbb{Z}^3, B)$ for short), where $B \subset \mathbb{Z}^3$ is the *foreground* and $B^c = \mathbb{Z}^3 \setminus B$ the *background*, is represented by the set of unit cubes (voxels) centered at the points of B together with all their faces. This identification of voxels with 3D cubes in \mathbb{R}^3 leads, in a natural way, to the combinatorial structure of cubical complexes, whose geometric building blocks (*cells*) are points, edges, squares and cubes (see [9]). More concretely, given a voxel centered at a point of \mathbb{Z}^3 of coordinates (i, j, k) , the cells associated to this voxel considered as a 3D cube are denoted as follows (see Fig 1):

- The eight vertices (0-cells): $(i \pm \frac{1}{2}, j \pm \frac{1}{2}, k \pm \frac{1}{2})$
- The twelve edges (1-cells): $(i, j \pm \frac{1}{2}, k \pm \frac{1}{2}), (i \pm \frac{1}{2}, j, k \pm \frac{1}{2}), (i \pm \frac{1}{2}, j \pm \frac{1}{2}, k)$.
- The six square faces (2-cells): $(i, j, k \pm \frac{1}{2}), (i, j \pm \frac{1}{2}, k), (i \pm \frac{1}{2}, j, k)$.
- The cube (3-cell): (i, j, k) .

By considering the (26, 6)-relationship we can guarantee that the topology of the cubical complex associated to the image reflects the topology of the object.

A cubical complex is, in fact, a special case of cell complex, which is a more general topological structure by which a space is decomposed into basic elements (cells) of different dimensions that are glued together by their boundaries.

Given a cell complex K , a *proper face* of $\sigma \in K$ is a face of σ whose dimension is strictly less than the one of σ . A *facet* of σ is a proper face of σ of maximal dimension. A *maximal cell* of K is a cell of K which is not a proper face of any other cell of K . The dimension of K is the maximal dimension of the maximal cell of K . A facet that is incident only to one cell is called a *boundary facet*. The union of all the boundary facets is the boundary of the cell complex which is denoted by ∂K .

The cubical complex $Q(I)$ representing a 3D binary digital image I satisfies that the maximal cells are cubes and the elements of the boundary of $Q(I)$, $\partial Q(I)$, are all the square faces of $Q(I)$ which are shared by a voxel of B and a voxel of B^c together with all their faces.

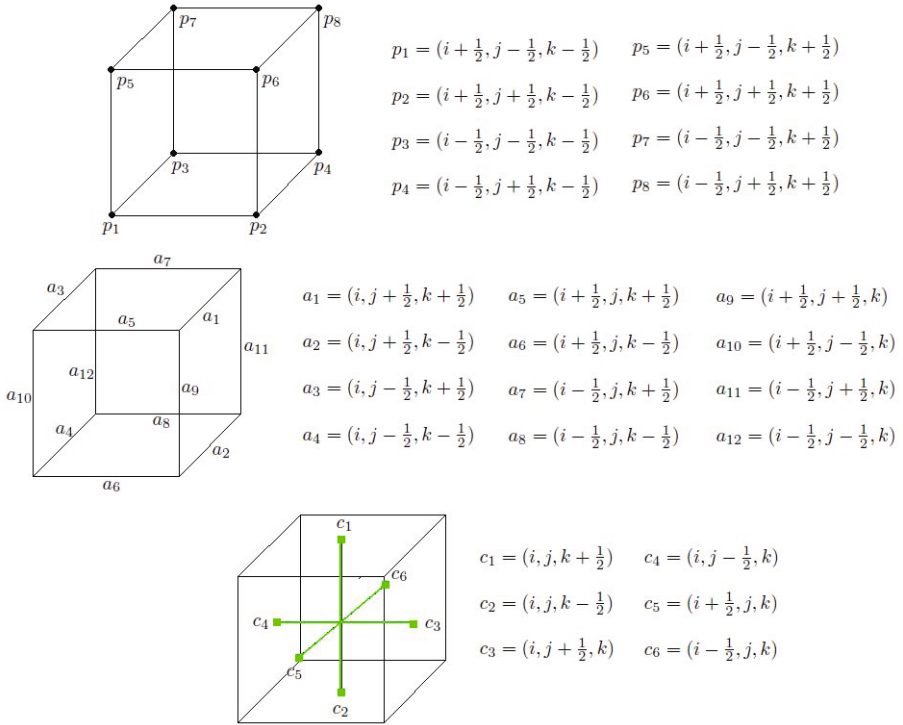


Fig. 1. Notation of the vertices $\{p_i\}$, edges $\{a_i\}$ and square faces $\{c_i\}$ associated to the voxel (i, j, k)

Given a cell complex K , define K_q as the set of q -cells of K . Define the morphism $k_q : K_q \times K_{q-1} \rightarrow \mathbb{Z}_2$ given by $k_q(c, c') := 1$ if c' is a facet of c and $k_q(c, c') := 0$ in other case (we do not take into account orientation). Then, we can codify a cell complex as a pair (K, k) where $K = \bigcup_q K_q$ is the set of cells of K and $k = \bigoplus_q k_q$ defines the relation between each cell and its facets. The morphism k is called the *incidence number* (see [14]).

For example, since any cubical complex Q is a particular case of cell complex, it can be codified as a pair (Q, k^Q) : Let Q_0 be the set of vertices, Q_1 the set of edges, Q_2 the set of square faces and Q_3 the set of cubes of Q . Define the morphism $k_q^Q : Q_q \times Q_{q-1} \rightarrow \mathbb{Z}_2$ given by $k_q^Q(c, c') := 1$ if c' is a facet of c and $k_q^Q(c, c') := 0$ in other case. Taking into account the notation given for the cells of Q as points in \mathbb{R}^3 , define $k_q^Q(c, c') := 1$ if the Euclidean distance between c and c' is $\frac{1}{2}$ and $k_q^Q(c, c') := 0$ in other case.

3 Well-Composed Cell Complexes

A 3D binary digital image $I = (\mathbb{Z}^3, B)$ is *well-composed* [10] if the boundary of the cubical complex associated, $\partial Q(I)$, is a 2D manifold, i.e. if each point in

$\partial Q(I)$ has a neighborhood homeomorphic to \mathbb{R}^2 . This definition implies a simple correspondence between a 3D binary digital image and the boundary surface of the associated cubical complex. Thus, one can use well-known properties of continuous boundary surfaces to determine and analyze properties of these digital images.

Since the boundary of the cubical complex associated to $I = (\mathbb{Z}^3, B)$ coincides with the one of $I^c = (\mathbb{Z}^3, B^c)$, a 3D binary digital image I is well-composed iff I^c is also well-composed.

3D binary digital images are often not well-composed images. Nevertheless, there are several methods for turning 3D binary digital images that are not well-composed into well-composed ones (see [12,15]). The problem is that, in general, these techniques do not guarantee the topological equivalence between the original object and its corresponding well-composed image, since they can modify the original image by moving some voxels from B^c to B .

In this section, we are interested in obtaining a homotopy-equivalent cell complex to the cubical complex associated to a 3D binary digital image whose geometric realization could enjoy the advantages of well-composed images. Specifically, we present a method to transform the cubical complex Q associated to a 3D binary digital image which is not generally a well-composed image into a cell complex (K, k) . This cell complex (K, k) satisfies that it is homotopy equivalent to Q and its boundary surface, ∂K , is composed by 2D manifolds.

The following proposition shows the characterization of well-composed images in terms of simple local conditions on cubes in the cubical complex associated, as one can observe in Fig. 2.

Proposition 1. [11] *A 3D binary digital image $I = (\mathbb{Z}^3, B)$ is well-composed iff the configurations of cubes $C1$, $C2$ and $C3$ (modulo reflections and rotations) do not occur in $Q(I)$:*

- C1** *Four cubes share an edge (a, b, c) and exactly two of them which do not share a face are contained in $Q(I)$ and the other two are not contained in $Q(I)$. That is, if W denotes the set of the four cubes sharing the edge (a, b, c) , there are exactly two cubes of W , denoted by w_1 and w_2 , that are cubes of $Q(I)$ and whose squared Euclidean distance is 2.*
- C2** *Eight cubes share a vertex (a, b, c) and exactly two of them which are corner-adjacent are contained in $Q(I)$ while the other six are not. That is, if S denotes the set of the eight cubes sharing the vertex (a, b, c) , there are exactly two cubes of S , denoted by s_1 and s_2 , that are cubes of $Q(I)$ and whose squared Euclidean distance is 3.*
- C3** *Eight cubes share a corner point and exactly two of them which are corner-adjacent are contained in $Q(I^c)$ while the other six are not. That is, if T denotes the set of the eight cubes sharing the vertex (a, b, c) . Then, there are exactly two cubes of T , denoted by t_1 and t_2 , that are not cubes of $Q(I)$ and whose squared Euclidean distance is 3.*

Given a cubical complex Q associated to a 3D binary digital image, we say that an edge (a, b, c) of Q is a *critical edge* if (a, b, c) is an edge shared by four

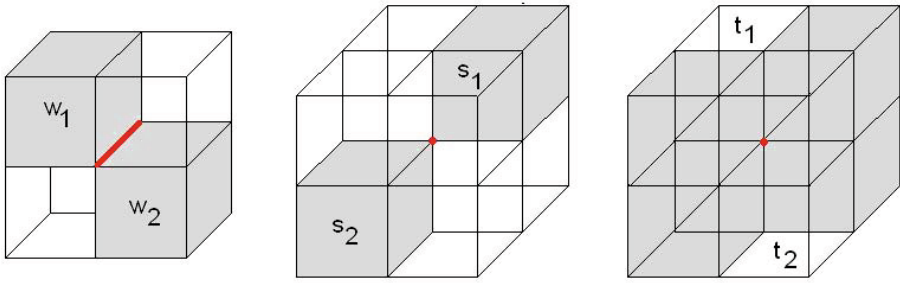


Fig. 2. Configurations C1, C2 and C3 (modulo reflections and rotations). These configurations cannot occur in $Q(I)$.

cubes that constitute the C1-configuration. We say that a vertex (a, b, c) of Q is a *critical vertex* if it is either a vertex shared by eight cubes that constitute the C2-configuration or a vertex shared by eight cubes that constitute the C3-configuration or a vertex shared by eight cubes such that one or several critical edges are incident to it. Summing up, one can observe in Fig. 3 all the critical configurations that are possible by the combination of the mentioned critical elements within a set of eight cubes sharing a vertex. Notice that configurations $C(2, 0)$, $C(2, 1)$, $C(3, 0)$ and $C(3, 1)$ are complementary to configurations $C(6, 2)$, $C(6, 1)$, $C(5, 1)$ and $C(5, 2)$, respectively; as well as configurations $C(4, 1)$, $C(4, 2)$ and $C(4, 3)$ are self-complementary.

Now, given a cubical complex Q associated to a 3D binary digital image, we present a method to generate a new cell complex by some basic operations on the input cubical complex to repair all the critical edges and critical vertices that appear in Q . The method consists of three steps: (1) all the critical edges and critical vertices that appear in Q are labeled and put into either the set E of critical edges or the set V of critical vertices; (2) apply Algorithm 1 to the cubical complex Q to repair all the edges of E and (3) apply Algorithm 2 to the cell complex output by the previous algorithm to repair all the vertices of V .

Algorithm 1. Repair the critical edges in E .

INPUT: The cubical complex (Q, k^Q) associated to a 3D binary digital image.

Initialize $K'_i := \{(a, b, c, 0) : \text{such that } (a, b, c) \in Q_i\}$ and $k'_i((a, b, c, 0), (a', b', c', 0)) := k_i^Q((a, b, c), (a', b', c'))$, for any $(a, b, c), (a', b', c') \in Q_i$ and for $0 \leq i \leq 3$.

- For each critical edge $(a, b, c) \in E$, do:
 - Duplicate the edge: $K'_1 := K'_1 \setminus \{(a, b, c, 0)\} \cup \{(a, b, c, 1), (a, b, c, 2)\}$.
 - Add a new 2-cell: $K'_2 := K'_2 \cup \{(a, b, c, 1, 2)\}$.
 - Denote by w_{11} and w_{12} the two square faces of w_1 sharing the edge (a, b, c) . Denote by w_{21} the one of the square faces of w_2 sharing the edge (a, b, c) such that the squared Euclidean distance between w_{11} and w_{21} is $\frac{1}{2}$. Denote by w_{22} the other square face of w_2 sharing the edge (a, b, c) (see Fig. 4).

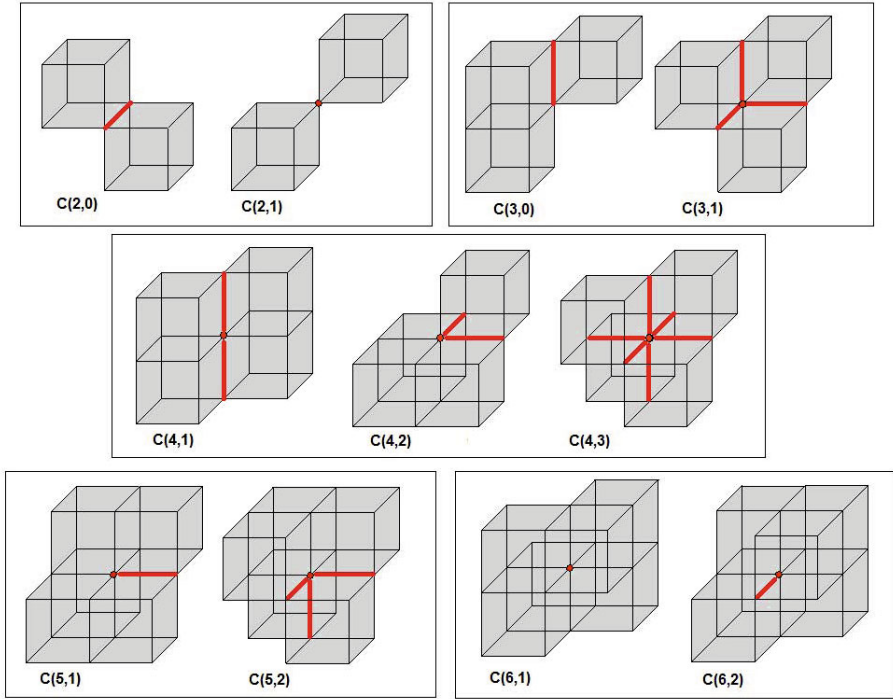


Fig. 3. All the critical configurations within a set of eight cubes sharing one vertex (modulo reflections and rotations)

- Define $k'_1((a, b, c, j), v) := k'_1((a, b, c, 0), v)$ for $j = 1, 2$ and for any vertex $v \in K'_0$.
- Define $k'_2(f, (a, b, c, j)) := 1$ if $f = w_{ij}$, for $i = 1, 2$, or $f = (a, b, c, 1, 2)$, for $j = 1, 2$.
- Define $k'_2((a, b, c, 1, 2), e) := 0$ if $e \neq (a, b, c, j)$ for $j = 1, 2$, that is, the only facets of the new 2-cell $(a, b, c, 1, 2)$ are the edges $(a, b, c, 1)$ and $(a, b, c, 2)$.
- Define $k'_3(w, (a, b, c, 1, 2)) := 1$ if $w = w_i$ for $i = 1, 2$ and 0 in other case.

OUTPUT: The cell complex (K', k') .

Algorithm 2. Repair the critical vertices in V .

INPUT: The cell complex (K', k') obtained after applying Algorithm 1.

Initialize $K_i := K'_i$ and $k_i := k'_i$ for $0 \leq i \leq 3$.

- For each critical vertex $(a, b, c) \in V$ with the configuration $C(2, 1)$, do:
 - Duplicate the vertex: $K_0 := K_0 \setminus \{(a, b, c, 0)\} \cup \{(a, b, c, 1), (a, b, c, 2)\}$.
 - Add two new edges: $K_1 := K_1 \cup \{(a, b, c, 1, 2), (a, b, c, 2, 1)\}$.
 - Add a new 2-cell: $K_2 := K_2 \cup \{(a, b, c, 1, 2, 1)\}$.

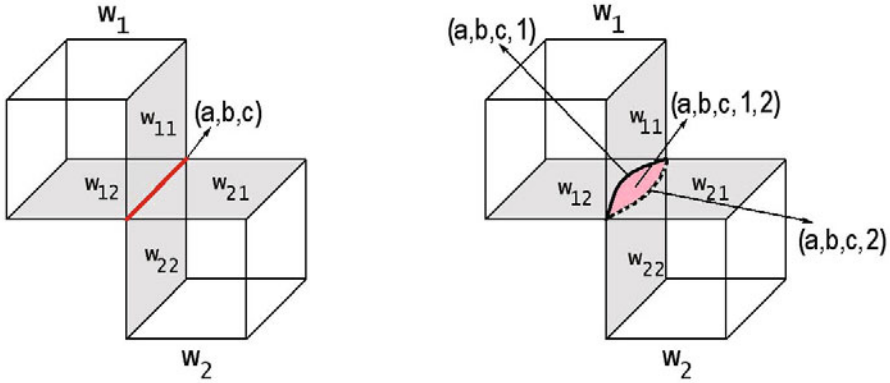


Fig. 4. Notations for the cells around a critical edge (a, b, c)

- Take two square faces of s_1 sharing the vertex (a, b, c) and denote them by s_{11} and s_{12} . Take the square face of s_2 sharing the vertex (a, b, c) whose squared Euclidean distance to s_{11} is 2; denote it by s_{22} . Denote by s_{21} the other square face of s_2 sharing the vertex (a, b, c) whose squared Euclidean distance to s_{12} is 2 (see Fig. 5).
 - Define $k_1(e, (a, b, c, 1)) := 1$ if either $e = (a, b, c, 1, 2)$ or $e = (a, b, c, 2, 1)$, or e is a face edge of s_1 shared by s_{11} and s_{12} or e is any of the other two edges of s_2 such that $k'_1(e, (a, b, c, 0)) = 1$ and that are not shared by s_{21} and s_{22} at the same time, and 0 otherwise.
 - Define $k_1(e, (a, b, c, 2)) := 1$ if either $e = (a, b, c, 1, 2)$ or $e = (a, b, c, 2, 1)$, or e is a face edge of s_2 shared by s_{21} and s_{22} or e is any of the other two edges of s_1 such that $k'_1(e, (a, b, c, 0)) = 1$ and that are not shared by s_{11} and s_{12} at the same time, and 0 otherwise.
 - Define $k_2(f, (a, b, c, 1, 2)) := 1$ if $f = s_{12}$ or $f = s_{22}$ or $f = (a, b, c, 1, 2, 1)$, and 0 otherwise.
 - Define $k_2(f, (a, b, c, 2, 1)) := 1$ if $f = s_{21}$ or $f = s_{11}$ or $f = (a, b, c, 1, 2, 1)$, and 0 otherwise.
 - Define $k_3(w, (a, b, c, 1, 2, 1)) := 1$ if $w = s_i$ for $i = 1, 2$ and 0 otherwise.
- For each critical vertex $(a, b, c) \in V$ with any configuration of Fig. 3 apart from $C(2, 1)$, do:
- Label the background connected components of the configuration of eight 3-cells around the critical vertex with labels 1, 2, 3 and 4, respectively (notice that the number of connected components may vary from 2 to 4). Let $L = \{1, 2, \dots, l\}$ be the set of labels assigned.
 - Label all the edges incident to (a, b, c) that belong to the boundary of K , with the corresponding label of the background component that shares such an edge.
 - Substitute the critical vertex by a set of vertices, one for each label corresponding to a background connected component: $K_0 := K_0 \setminus \{(a, b, c, 0)\} \cup \{(a, b, c, 1), \dots, (a, b, c, l)\}$.

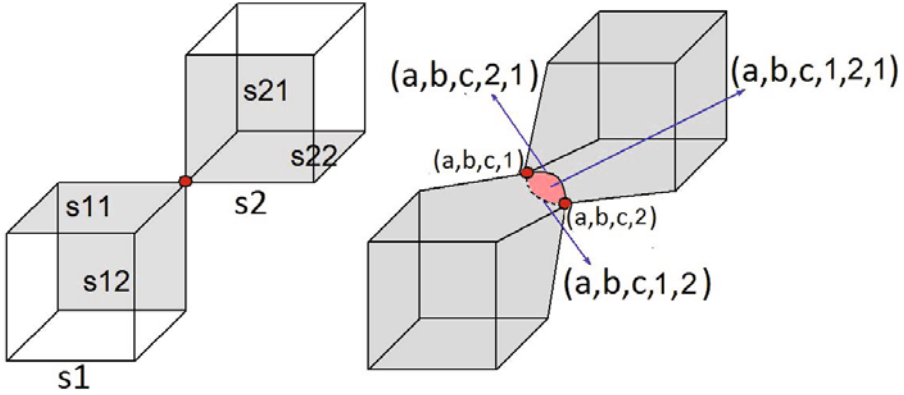


Fig. 5. Notations for the cells around a critical vertex (a, b, c) with the configuration $C(2, 1)$

- Add new edges: $K_1 := K_1 \cup \{(a, b, c, i, j), i, j \in L, i < j\}$.
- If $l \geq 3$, add new 2-cells $K_2 := K_2 \cup \{(a, b, c, i, j, k), i, j, k \in L, i < j < k\}$.
- If $l = 4$, add a new 3-cell $K_3 := K_3 \cup \{(a, b, c, 1, 2, 3, 4)\}$.
- Define $k_1(e, (a, b, c, i)) := 1$ if either $k'_1(e, (a, b, c, 0)) = 1$ and e is an edge with label i or $e = (a, b, c, i, j)$, for any j or $e = (a, b, c, j, i)$, for any j , and 0 otherwise.
- Define $k_2(f, (a, b, c, i, j)) := 1$ if either f is a 2-cell with two face edges labeled as i and j or $f = (a, b, c, i, j, k)$ or $f = (a, b, c, i, k, j)$ or $f = (a, b, c, k, i, j)$, and 0 otherwise.
- Define $k_3(w, (a, b, c, i, j, k)) := 1$ if either w is a 3-cell with three face edges labeled as i, j and k , or $w = (a, b, c, 1, 2, 3, 4)$, and 0 otherwise.

OUTPUT: The cell complex (K, k) .

Proposition 2. The cell complex (K, k) and the cubical complex Q are homotopy equivalent. Moreover, the boundary surface ∂K , of the cell complex (K, k) , is composed by 2D manifolds, that is, each point of ∂K has a neighborhood homeomorphic to \mathbb{R}^2 .

Example 1. Consider the 3D binary digital image $I = (\mathbb{Z}^3, B)$ with $B = \{(1, 1, 0), (0, 0, 1), (0, 2, 1), (0, 1, 2)\}$. Let Q the cubical complex associated to the image. This cubical complex has 4 voxels (3-cells), 24 square faces (2-cells), 45 edges (1-cells) and 26 vertices (0-cells). The conflictive cells of K are:

- The edges $a_1 = (0, \frac{1}{2}, \frac{3}{2})$ and $a_2 = (0, \frac{3}{2}, \frac{3}{2})$;
- The vertices $v_1 = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ and $v_2 = (\frac{1}{2}, \frac{3}{2}, \frac{1}{2})$.

Applying the previous method to this cubical complex, the obtained well-composed cell complex K have four 3-cells, twenty eight 2-cells (21 square faces, 2 pentagons and 1 hexagon), 51 edges and 28 vertices (see Fig. 7).

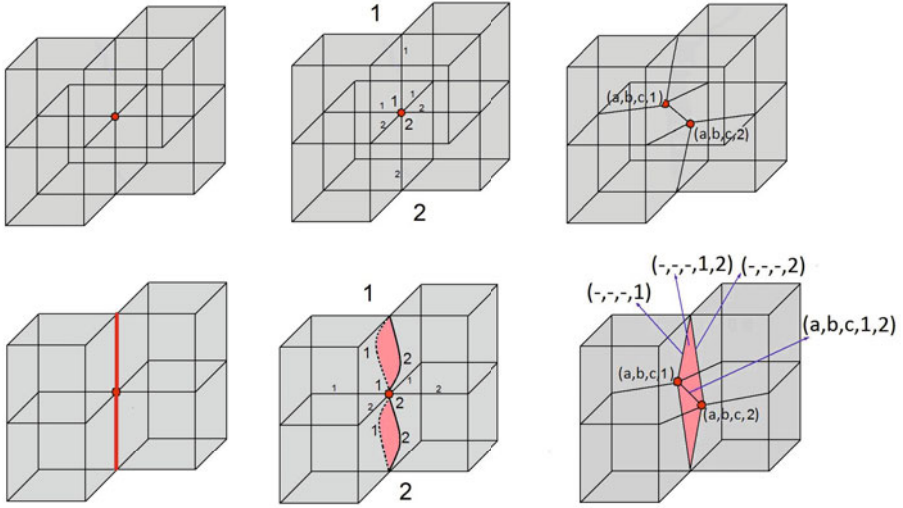


Fig. 6. Notations for the cells around a critical vertex (a, b, c) , with the configurations $C(6, 1)$ (first row) and $C(4, 1)$ (second one)

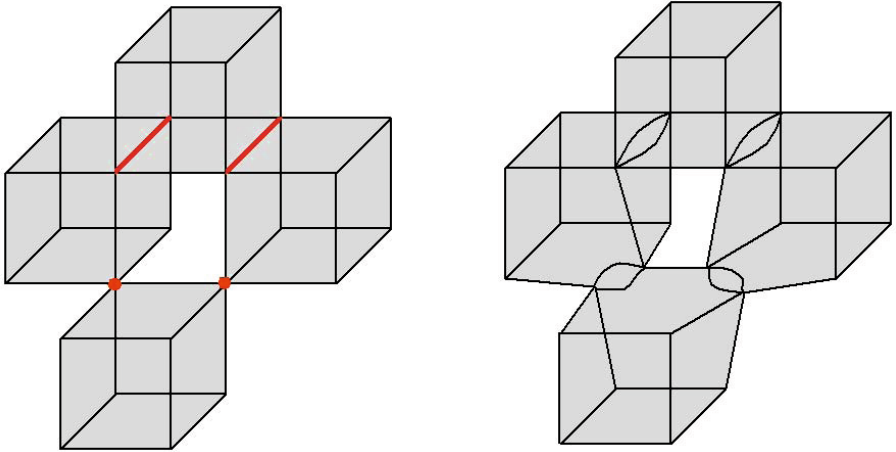


Fig. 7. Example of a cubical complex and well-composed complex

4 Conclusion and Future Work

We have presented a method for obtaining a well-composed cell complex from the cubical complex associated to a 3D binary digital image. We are convinced that this new representation will satisfy very nice properties: first, the new cell complex will allow to compute the homology of the image by computing the

homology of the boundary surface of the cell complex; we will be able to geometrically control the representative cycles of homology generators in the sense that, for example, a 1-cycle will never belong to two different cavities (in fact, this could simplify the computation of the cup product in cohomology); moreover, algorithms developed for surfaces embedded in \mathbb{R}^3 ([1,2,3]) could be applied to the well-composed cell complex. Another future plan is to improve the notation used here to define the incidence index k for the new cells.

References

1. Dey, T.K., Li, K.: Persistence-based handle and tunnel loops computation revisited for speed up. *Computers & Graphics* 33, 351–358 (2009)
2. Dey, T.K., Li, K., Sun, J.: On computing handle and tunnel loops. In: *IEEE Proceedings of the International Conference on Cyberworlds*, pp. 357–366 (2007)
3. Dey, T.K., Li, K., Sun, J., Cohen-Steiner, D.: Computing Geometry-aware Handle and Tunnel Loops in 3D Models. *ACM Transactions on Graphics* 27(3), Article 45 (2008)
4. Gonzalez-Diaz, R., Jimenez, M.J., Medrano, B.: Cohomology ring of 3D cubical complexes. In: *Proc. of the 13th International Workshop on Combinatorial Image Analysis, IWCIA 2009. Progress in Combinatorial Image Analysis*, pp. 139–150 (2009)
5. Gonzalez-Diaz, R., Real, P.: Towards Digital Cohomology. In: Nyström, I., Sanniti di Baja, G., Svensson, S. (eds.) *DGCI 2003. LNCS*, vol. 2886, pp. 92–101. Springer, Heidelberg (2003)
6. Gonzalez-Diaz, R., Real, P.: On the Cohomology of 3D Digital Images. *Discrete Applied Math.* 147, 245–263 (2005)
7. Gonzalez-Diaz, R., Jimenez, M.J., Medrano, B.: Cubical cohomology ring of 3D pictures. *International Journal of Imaging Systems and Technologies* (to appear)
8. Krahnstoever, N., Lorenz, C.: Computing curvature-adaptive surface triangulations of three-dimensional image data. *Vis. Comput.* 20(1), 17–36 (2004)
9. Kaczynski, T., Mischaikow, K., Mrozek, M.: *Computational Homology*. Applied Mathematical Sciences 157 (2004)
10. Latecki, L.J., Eckhardt, U., Rosenfeld, A.: Well-composed Sets. *Comput. Vis. Image Underst.* 61(1), 70–83 (1995)
11. Latecki, L.J.: 3D Well-Composed Pictures. *Graphical Models and Image Processing* 59(3), 164–172 (1997)
12. Latecki, L.: *Discrete Representation of Spatial Objects in Computer Vision*. Kluwer Academic, Dordrecht (1998)
13. Marchadier, J., Arques, D., Michelin, S.: Thinning grayscale well-composed images. *Pattern Recognit. Lett.* 25(5), 581–590 (2004)
14. Massey, W.S.: *A basic course in algebraic topology*. Springer, New York (1991)
15. Siqueira, M., Latecki, L.J., Tustison, N., Gallier, J., Gee, J.: Topological Repairing of 3D Digital Images. *J. Math. Imaging Vis.* 30, 249–274 (2008)

A Unified Topological Framework for Digital Imaging

Loïc Mazo^{1,2,*}, Nicolas Passat¹, Michel Couprie², and Christian Ronse¹

¹ Université de Strasbourg, LSIIT, UMR CNRS 7005, France

² Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge, Équipe A3SI,
ESIEE Paris, France

{loic.mazo, passat, cronse}@unistra.fr, m.couprie@esiee.fr

Abstract. In this article, a tractable *modus operandi* is proposed to model a (binary) digital image (*i.e.*, an image defined on \mathbb{Z}^n and equipped with a standard pair of adjacencies) as an image defined in the space of cubical complexes (\mathbb{F}^n). In particular, it is shown that all the standard pairs of adjacencies in \mathbb{Z}^n can then be correctly modelled in \mathbb{F}^n . Moreover, it is established that the digital fundamental group of a digital image in \mathbb{Z}^n is isomorphic to the fundamental group of its corresponding image in \mathbb{F}^n , thus proving the topological correctness of the proposed approach. From these results, it becomes possible to establish links between topology-oriented methods developed either in classical digital spaces (\mathbb{Z}^n) or cubical complexes (\mathbb{F}^n).

Keywords: digital imaging, digital topology, cubical complexes, homotopy, fundamental group.

1 Introduction

The rise of digital imaging, and the associated need of efficient image analysis tools, have provided a strong motivation to research related to the definition of sound digital topological models. Indeed, in order to process digital images, it is often fundamental to be able to preserve, get back or integrate topological information.

Basically, an n -dimensional (digital) binary image can be considered as a subset of \mathbb{Z}^n . However, the actual structures visualised in such images are generally continuous ones, corresponding to objects of the real world, *i.e.*, objects of \mathbb{R}^n , and not of \mathbb{Z}^n . In order to deal with this continuous/discrete issue, research efforts have essentially focused on specific and pragmatic questions related to topology, namely the definition of a notion of *adjacency* relation, and the induced notions of *connectivity* and *arcs*. These notions lead, in particular, to high-level concepts of topology, such as *homotopy*, with natural applications to “homotopy type-preserving” image processing.

The first solution proposed to model the topology of a digital image in \mathbb{Z}^n was to consider that two points (also called *xels*) are adjacent if they are neighbours in the n -D cubic grid naturally induced by \mathbb{Z}^n . In this framework, partial solutions have been found to model as well as possible the above topological properties, for instance by defining *dual adjacencies* for the object and the background (composed of the xels of value

* The research leading to these results has received funding from the French *Agence Nationale de la Recherche* (Grant Agreement ANR-2010-BLAN-0205).

1 and 0, respectively), enabling to define, from these adjacency relations, the notions of connectivity [19] and of *digital fundamental group* [10], which permits to compare objects from a topological point of view. This approach is known as *digital topology* [13].

Other discrete spaces, enabling to model the continuous topological properties of digital images, have also been proposed. These alternative approaches of topology modelling are *connected ordered topological spaces* [9], *abstract cell complexes* [14] and *orders* [4]. Broadly speaking, they propose to put some “topological glue” between the xels of digital images to define the topological links with their continuous analogues.

By comparison to these (more sophisticated) approaches, digital topology may appear as the less satisfactory solution to deal with topological properties of binary images. Nevertheless, it remains the most commonly used framework for developing image processing tools dealing with topological issues. Indeed, since digital topology is directly defined in \mathbb{Z}^n , methods relying on it will also provide final results in \mathbb{Z}^n , which is a desired property in most applications. Moreover, a large literature has been devoted to homotopy-type preservation in digital topology, especially thanks to the concept of simple point [6]. In this context, very few methods have been based on alternative models while digital topology has led to the design of quite numerous ones (see, e.g., [117]).

Because of this intensive use of digital topology, it may be important to guarantee that there exists an actual compatibility between digital topology and the other proposed topological approaches (and more generally with the “continuous” topology). This requires to be able to embed a binary image defined in \mathbb{Z}^n into a richer space (while respecting the chosen adjacencies in \mathbb{Z}^n) while preserving certain topological characteristics of objects (see e.g. [11][16][15][8]).

The “richer space” that is used here is \mathbb{F}^n , namely the space of cubical complexes, which is together a connected ordered topological space, a cellular space and an order (i.e., a poset). Though it is commonly admitted that there exists a strong link between digital topology and cubical complexes [12], since complexes are closed objects, the images handled in \mathbb{F}^n correspond generally to images defined in \mathbb{Z}^n with a $(3^n - 1, 2n)$ -adjacency pair. In [5], a method has been proposed to retrieve and improve digital topology in the framework of posets, but the case of the (6, 18)- and (18, 6)-adjacency pairs was not considered. In [3], a way is described to embed digital pictures in a space of complexes according to the kind of connectivity chosen in \mathbb{Z}^n . However, there is no use of an intrinsic topology on complexes which are just a step between \mathbb{Z}^n and \mathbb{R}^n , leading to define specific notions of connectedness and digital homotopy in \mathbb{F}^n . Thereby, in this paper, we propose a complete framework to correctly embed a binary digital image in the topological space \mathbb{F}^n , according to the choice of adjacencies which has been made in \mathbb{Z}^n .

The article is organised as follows. Sec. 2 recalls background notions. Sec. 3 describes the mapping enabling to associate a binary image defined on \mathbb{Z}^n to an equivalent image defined in \mathbb{F}^n . Sec. 4 presents the main contribution of this work. It states that the connected components and the digital fundamental group of the digital images in \mathbb{Z}^n are preserved in \mathbb{F}^n when using the mapping described in Sec. 3. Sec. 5 concludes this article. By lack of space, no proofs are given in this article, they can be found in [17].

2 Background Notions

This section provides the minimal set of background notions required to make this article globally self-contained, and then more comprehensible for the reader.

2.1 Posets

A *partially ordered set* (or *poset*) is a ordered pair (X, \leq) where X is a set and the relation \leq is a partial order on X . We write $x < y$ when $x \leq y$ and $x \neq y$. The relation \geq defined on X by $x \geq y$ if $y \leq x$ is a partial order on X called the *dual order*. The *covering* relation \prec , associated to \leq , is defined by: $x \prec y$ (say “ y covers x ”) if $x < y$ and there is no z such that $x < z < y$. We set: $x^\uparrow = \{y \in X \mid x \leq y\}$; $x^{\uparrow+} = \{y \in x^\uparrow \mid y^\uparrow = \{y\}\}$; $x^\downarrow = \{y \in X \mid y \leq x\}$; $x^{\downarrow*} = x^\downarrow \setminus \{x\} = \{y \in X \mid y < x\}$; $x^\prec = \{y \in X \mid x \prec y\}$. An element $x \in X$ is *minimal* if $x^\downarrow = \{x\}$ and *maximal* if $x^\uparrow = \{x\}$. An element $x \in X$ is the *minimum* of X if $x^\uparrow = X$ and is the *maximum* of X if $x^\downarrow = X$. If the minimum (resp., the maximum) exists, then it is unique.

2.2 Topological Spaces

Let (X, \leq) be a poset. The set \mathcal{U} defined by $\mathcal{U} = \{U \subseteq X \mid \forall x \in U, x^\uparrow \subseteq U\}$ is a topology on X which is called the *Alexandroff topology*. In this topology, the set x^\uparrow is the smallest open set containing x (or the smallest neighbourhood of x , called the *star* of x) and the set x^\downarrow is the smallest closed set containing x (the *closure* of x). Two points $x, y \in X$ are *adjacent* if $x \leq y$ or $y \leq x$. A sequence $(z_i)_{i=0}^r$ ($r \geq 0$) of elements of X is an *arc* in X (from z_0 to z_r) if for all $i \in \llbracket 1, r \rrbracket$ ¹, z_{i-1} and z_i are distinct and adjacent. A subset Y of X is *connected* if for all $x, y \in Y$, there exists an arc in Y from x to y . A *connected component* of a subset Y of X is a maximal (for inclusion) connected subset of Y .

The *closure* Y^\downarrow of a subset $Y \subseteq X$ is the smallest closed set including Y . The *interior* Y° of a subset $Y \subseteq X$ is the largest open set included in Y . Closure and interior are dual notions since $\neg(Y^\circ) = (\neg Y)^\downarrow$ and $\neg(Y^\downarrow) = (\neg Y)^\circ$ where $\neg Y = X \setminus Y$. An open set Y is a *regular open set* if $Y = (Y^\downarrow)^\circ$ and a closed set is a *regular closed set* if $Y = (Y^\circ)^\downarrow$. The complement of a regular open set is a regular closed set.

2.3 Cubical Complexes

Let \mathbb{Z} be the set of integers. Let $\mathbb{F}_0^1 = \{\{a\} \mid a \in \mathbb{Z}\}$ and $\mathbb{F}_1^1 = \{\{a, a + 1\} \mid a \in \mathbb{Z}\}$. Let $n \geq 1$.

Let $f \subseteq \mathbb{Z}^n$. If f is the Cartesian product of m elements of \mathbb{F}_1^1 and $n - m$ elements of \mathbb{F}_0^1 , we say that f is a *face* or an *m-face* (of \mathbb{Z}^n), m is the *dimension* of f , and we write $\dim(f) = m$ (some faces of \mathbb{Z}^2 are depicted in Fig. 1). We denote by \mathbb{F}^n the set composed of all faces of \mathbb{Z}^n ; this set is the (n -D) space of *cubical complexes*. We denote by \mathbb{F}_k^n ($0 \leq k \leq n$) the set composed of all k -faces of \mathbb{Z}^n . The couple $(\mathbb{F}^n, \subseteq)$ is a poset. Let $F \subseteq \mathbb{F}^n$ be a set of faces. Let $f \in F$ be a face. The face f is a *facet* of F if f is maximal in F . In particular, if $x = (x_i)_{i=1}^n \in \mathbb{Z}^n$, the set $\dot{x} = \prod_{i=1}^n \{x_i, x_i + 1\}$ is a facet of \mathbb{F}^n .

¹ We write $\llbracket p, q \rrbracket$ for an integer interval and $[p, q]$ for a real interval.

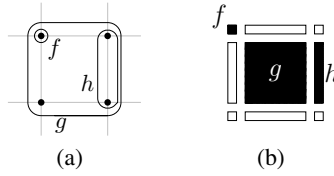


Fig. 1. Two representations of a set of faces $F = \{f, g, h\}$ in \mathbb{Z}^2 with, for instance, $f = \{0\} \times \{1\}$, $g = \{0, 1\} \times \{0, 1\}$ and $h = \{1\} \times \{0, 1\}$. The face g is a facet of F (and also of \mathbb{F}^2).

2.4 Digital Topology

Let $x = (x_i)_{i=1}^n$ and $y = (y_i)_{i=1}^n$ be two points in \mathbb{Z}^n (also called n -xels, or simply xels). The points x and y are $2n$ -adjacent if $\sum_{i=1}^n |x_i - y_i| = 1$. They are $(3^n - 1)$ -adjacent if $\max_{i=1}^n |x_i - y_i| = 1$. When $n = 3$, the points x and y are 18 -adjacent if they are 26 -adjacent and $\sum_{i=1}^n |x_i - y_i| \leq 2$. Let $\alpha \in \{2n, 3^n - 1\}$ (or possibly $\alpha = 18$ if $n = 3$). Any point in \mathbb{Z}^n is α -adjacent to α other points. A sequence $\gamma = (z_i)_{i=0}^r$ ($r \geq 0$) of points in $X \subseteq \mathbb{Z}^n$ is a (digital) α -path (from z_0 to z_r) if for all $i \in \llbracket 1, r \rrbracket$, z_{i-1} and z_i are α -adjacent. The integer r is the length of γ . A subset $X \subseteq \mathbb{Z}^n$ is α -connected, if for all $x, y \in X$, there exists a digital α -path from x to y in X . In order to retrieve some topological features in binary digital images (such as the notion of hole), it is necessary to use pairs of adjacencies, one for the object X and one for the background $\mathbb{Z}^n \setminus X$. The suitable pairs are $(2n, 3^n - 1)$ and $(3^n - 1, 2n)$ (plus, when $n = 3$, $(6, 18)$ and $(18, 6)$).

3 Connectivity: From \mathbb{Z}^n to \mathbb{F}^n

A (digital) image λ on \mathbb{Z}^n is a function from \mathbb{Z}^n to $\{0, 1\}$. A (complex) image μ on \mathbb{F}^n is a function from \mathbb{F}^n to $\{0, 1\}$. The object (resp. the background) associated to the image $\theta : X \rightarrow \{0, 1\}$ (with $X = \mathbb{Z}^n$ or \mathbb{F}^n) is the set $\theta^{-1}(\{1\})$ (resp. $\theta^{-1}(\{0\})$).

If μ is a complex image, we write $\bigvee_{x \in X} \mu(x)$ (resp. $\bigwedge_{x \in X} \mu(x)$) for the maximum (resp. minimum) of the set $\{\mu(x) \mid x \in X\}$ and we also write $\mu(a) \vee \mu(b)$ (resp. $\mu(a) \wedge \mu(b)$) for $\bigvee_{x \in \{a,b\}} \mu(x)$ (resp. $\bigwedge_{x \in \{a,b\}} \mu(x)$).

The poset $(\mathbb{F}^n, \subseteq)$ is equipped with its Alexandroff topology.

3.1 One-to-One Correspondence between Images on \mathbb{Z}^n and \mathbb{F}^n

When two faces $g, h \in \mathbb{F}^n$ cover a face $f \in \mathbb{F}^n$ and their smallest neighbourhoods do not intersect (i.e., $g^\uparrow \cap h^\uparrow = \emptyset$), we say that they are opposite with respect to the face f . We denote $\text{opp}(f)$ the set of all $\{g, h\}$ for g opposite to h w.r.t. f . Intuitively, the face f is required to “locally connect” the faces g and h . When f is a facet, we have $\text{opp}(f) = \emptyset$.

Definition 1. (regular image) Let $\varepsilon : \llbracket 1, n \rrbracket \rightarrow \{-1, 1\}$ be a function called connectivity function [2]. A function $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ is an ε -regular image (or simply a regular image) if for all $m \in \llbracket 1, n \rrbracket$ and $f \in \mathbb{F}^n_{m-1}$, we have, recursively

² We write $\varepsilon = (a_1, \dots, a_n)$ for the function ε such that $\varepsilon(i) = a_i$ for all $i \in \llbracket 1, n \rrbracket$, as done for instance in Fig. 2. We also write $\varepsilon = 1$ (or $\varepsilon = -1$) when ε is a constant function.

$$\mu(f) = \begin{cases} \bigwedge_{\{a,b\} \in \text{opp}(f)} \mu(a) \vee \mu(b) & \text{if } \varepsilon(m) = 1 \\ \bigvee_{\{a,b\} \in \text{opp}(f)} \mu(a) \wedge \mu(b) & \text{if } \varepsilon(m) = -1 \end{cases}$$

For each connectivity function $\varepsilon : \llbracket 1, n \rrbracket \rightarrow \{-1, 1\}$, we define the function $\zeta_\varepsilon : \{0, 1\}^{\mathbb{Z}^n} \rightarrow \{0, 1\}^{\mathbb{F}^n}$ which maps any digital image λ to the unique ε -regular image $\zeta_\varepsilon(\lambda)$ such that, for each $a \in \mathbb{Z}^n$, we have $\zeta_\varepsilon(\lambda)(a) = \lambda(a)$. It is obvious that, for each ε , the function ζ_ε is a bijection between the set of digital images $\{0, 1\}^{\mathbb{Z}^n}$ and the subset of ε -regular images of $\{0, 1\}^{\mathbb{F}^n}$. Moreover, thanks to the choice of the connectivity function ε , we can accurately “carve” an image in \mathbb{F}^n to model the desired connectivity in \mathbb{Z}^n (see Fig. 2). In particular, we can get the usual pairs of adjacencies (see Figs. 2-4 and Table 1). In Section 4, the correspondences given in Table 1 will be justified by two theorems establishing that, by following these links, we preserve the connected components and fundamental groups.

When the function ε is constant, Definition 1 can be simplified. Note that the case $\varepsilon = -1$ corresponds to the $2n$ -adjacency in \mathbb{Z}^n while the case $\varepsilon = 1$ corresponds to the $(3^n - 1)$ -adjacency in \mathbb{Z}^n .

Proposition 1. *Let $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ be an ε -regular image. Let f be a face of \mathbb{F}^n .*

- (i) *If $\forall m > \dim(f), \varepsilon(m) = -1$, then we have $\mu(f) = \bigwedge_{f < a} \mu(a) = \bigwedge_{a \in f^+} \mu(a)$*
- (ii) *If $\forall m > \dim(f), \varepsilon(m) = 1$, then we have $\mu(f) = \bigvee_{f < a} \mu(a) = \bigvee_{a \in f^+} \mu(a)$*

In particular, if $\varepsilon = -1$ then $\mu(f) = \bigwedge_{a \in f^+} \mu(a)$ for all $f \in \mathbb{F}^n$, and if $\varepsilon = 1$ then $\mu(f) = \bigvee_{a \in f^+} \mu(a)$ for all $f \in \mathbb{F}^n$.

3.2 Duality

Let $\theta : X \rightarrow \{0, 1\}$ with $(X = \mathbb{Z}^n \text{ or } \mathbb{F}^n)$ be an image. We define the *negative image* $\neg\theta : X \rightarrow \{0, 1\}$ of θ by $\neg\theta(x) = 1 - \theta(x)$, for all $x \in X$.

Proposition 2. *If $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ is an ε -regular image, then $\neg\mu$ is $(-\varepsilon)$ -regular.*

Let $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ be an ε -regular image. We define the image $-\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ by $(-\mu)(f) = \mu(f)$ for all $f \in \mathbb{F}^n$ and μ is $(-\varepsilon)$ -regular.

Proposition 3. *If $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ is an ε -regular image, then we have $\neg(-\mu) = -(\neg\mu)$.*

From the above definitions and propositions, we straightforwardly derive the following result.

Proposition 4. *Let $\lambda : \mathbb{Z}^n \rightarrow \{0, 1\}$ be a digital image. Let $\varepsilon : \llbracket 1, n \rrbracket \rightarrow \{-1, 1\}$ be a connectivity function. Then, we have $\neg(\zeta_\varepsilon(\lambda)) = \zeta_{-\varepsilon}(\neg\lambda)$.*

$$\begin{array}{ccc} \lambda & \xrightarrow{\neg} & \neg\lambda \\ \zeta_\varepsilon \downarrow & & \downarrow \zeta_{-\varepsilon} \\ \zeta_\varepsilon(\lambda) & \xrightarrow{\neg} & \zeta_{-\varepsilon}(\neg\lambda) \end{array}$$

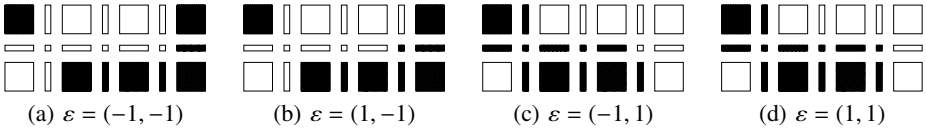


Fig. 2. Images $\zeta_\varepsilon(\lambda) : \mathbb{F}^2 \rightarrow \{0, 1\}$ for some given $\lambda : \mathbb{Z}^2 \rightarrow \{0, 1\}$

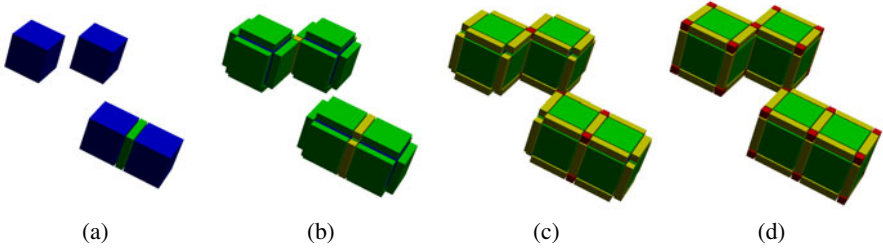


Fig. 3. Images $\zeta_\varepsilon(\lambda) : \mathbb{F}^3 \rightarrow \{0, 1\}$ for some given $\lambda : \mathbb{Z}^3 \rightarrow \{0, 1\}$. In Figs. 3-4 the different colours are only used to distinguish the faces of the object $(\zeta_\varepsilon(\lambda))^{-1}(\{1\})$ (blue: 3-faces; green: 2-faces; yellow: 1-faces; red: 0-faces). (a) With $\varepsilon(3) = -1$ (whatever values for $\varepsilon(1)$ and $\varepsilon(2)$), we obtain the 6-adjacency (for the object) in \mathbb{Z}^3 . (b) With $\varepsilon(3) = 1$ and $\varepsilon(2) = -1$, we obtain the 18-adjacency in \mathbb{Z}^3 . (c, d) With $\varepsilon(3) = \varepsilon(2) = 1$, we obtain the 26-adjacency in \mathbb{Z}^3 .

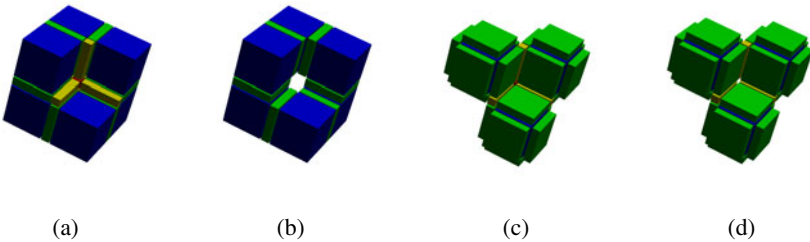


Fig. 4. A torus built with six 3-faces illustrates how the two 6-adjacencies can be obtained. (a) With $\varepsilon = (\pm 1, 1, -1)$, the foreground is a horn-torus so we obtain the 6-adjacency associated to the 18-adjacency. (b) With $\varepsilon = (\pm 1, -1, -1)$, the foreground is a ring-torus, so we obtain the 6-adjacency associated to the 26-adjacency. (c,d) An object built from three facets with two connectivity functions which could *a priori* be used to model the 18-adjacency (see Fig. 3(b)). In (c), with $\varepsilon = (1, -1, 1)$, we can see a red 0-face between the three cubes. This is what is expected for the background must have a 6-adjacency. In (d), with $\varepsilon = (-1, -1, 1)$, there is a hole instead of the red 0-face, which is not correct in 18-adjacency.

Table 1. Correspondence between pairs of adjacencies in \mathbb{Z}^n and connectivity functions

Space dimension	$n = 2$		$n = 3$				$n \geq 4$ (actually, $n \in \mathbb{N}^*$)	
Adjacencies in \mathbb{Z}^n	(4, 8)	(8, 4)	(6, 26)	(6, 18)	(18, 6)	(26, 6)	$(2n, 3^n - 1)$	$(3^n - 1, 2n)$
ε	-1	1	-1	$(-1, 1, -1)$	$(1, -1, 1)$	1	-1	1

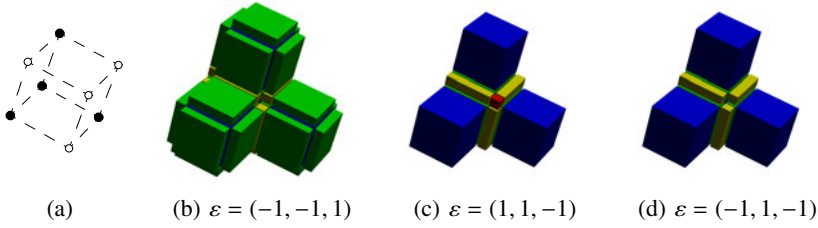


Fig. 5. (a) Symbolic representation of a trihedron related to a face $f \in \mathbb{F}^n$ such that $\dim(f) = n - 3$. Black dots: $f_1^{\uparrow+}$; white dots $f^{\uparrow+} \setminus f_1^{\uparrow+}$. The dash lines represent the existence of a face of dimension $n - 1$ forming the intersection between two facets of $f^{\uparrow+}$. (b-d) Examples of trihedra, with three connectivity functions ε (one of the blue facets is hidden).

Remark 1. This proposition establishes that, for a given connectivity function ε (and the associated pair of adjacencies (α, β)), all the properties valid for $\lambda^{-1}(\{1\})$ and $\mu^{-1}(\{1\})$ are also valid for $\lambda^{-1}(\{0\})$ and $\mu^{-1}(\{0\})$ for the opposite connectivity function $-\varepsilon$ (and the associated pair of adjacencies (β, α)). Broadly speaking, this means that the notions of object and background can be switched without loss of generality, provided that the pair of adjacencies (β, α) is also switched accordingly.

3.3 Computing Values Directly from Facets

The aim of this section is to find the number of facets which must have the value 1 in the star of a face to ensure that this face also has value 1. In \mathbb{F}^2 , the answer is straightforward. In \mathbb{F}^3 , it requires to carefully study a particular configuration (depicted in Fig. 5), however, it can be answered, as stated hereafter. In higher dimensional spaces, the particular configurations to study are too numerous to get a useful result.

Let $f \in \mathbb{F}^n$, with $n \geq 3$. If $\dim(f) = n - 3$, the poset $(f^{\uparrow}, \subseteq)$ has a unique minimum, namely f , and 8 maximal elements, namely the facets forming $f^{\uparrow+}$. From an adjacency point of view, these facets are geometrically organised as the 8 vertices of a cubical structure. When $f_1^{\uparrow+}$ (i.e., the facets of $f^{\uparrow+}$ whose values are equal to 1) is organised as in the configuration depicted in Fig. 5(a) (up to rotations and symmetries), we say that $f_1^{\uparrow+}$ is a *trihedron*. We define $\text{Card}^-(E) = 3$ and $\text{Card}^+(E) = 5$, if E is a trihedron, and $\text{Card}^-(E) = \text{Card}^+(E) = \text{Card}(E)$ otherwise.

For each connectivity function ε , we define recursively the function $\delta_\varepsilon : \llbracket 0, n \rrbracket \rightarrow \llbracket 1, 2^n \rrbracket$ by $\delta_\varepsilon(0) = 1$, and for all $i > 0$, $\delta_\varepsilon(i + 1) = 2\delta_\varepsilon(i) - 1$ if $\varepsilon(n - i) = 1$, and $\delta_\varepsilon(i + 1) = 2\delta_\varepsilon(i)$ if $\varepsilon(n - i) = -1$. It is easy to check that, for all $m \in \llbracket 0, n \rrbracket$, we have $\delta_\varepsilon(m) = 1 + \sum_{k=1}^m (1 - \varepsilon(n - k + 1))2^{m-k-1}$.

In the sequel, we write $f_1^{\uparrow+}$ for the set of facets in the star of f which have value 1: $f_1^{\uparrow+} = \{g \in \mathbb{F}_n^{\uparrow} \mid f \leq g \text{ and } \mu(g) = 1\}$.

Proposition 5. *Let $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ be an ε -regular image. Let f be a k -face of \mathbb{F}^n (with $n - 3 \leq k \leq n - 1$).*

- (i) *If $\text{Card}^-(f_1^{\uparrow+}) \geq \delta_\varepsilon(n - k)$, then $\mu(f) = 1$.*
- (ii) *If $\mu(f) = 1$, then $\text{Card}^+(f_1^{\uparrow+}) \geq \delta_\varepsilon(n - k)$.*

Table 2. Necessary and sufficient conditions to obtain $\mu(f) = 1$ (see Corollary [11](#))

dim(f) = $n - 2$				
ε	$(\dots, 1, 1)$	$(\dots, -1, 1)$	$(\dots, 1, -1)$	$(\dots, -1, -1)$
C_ε	Card($f_1^{\uparrow+}$) ≥ 1	Card($f_1^{\uparrow+}$) ≥ 2	Card($f_1^{\uparrow+}$) ≥ 3	Card($f_1^{\uparrow+}$) ≥ 4

dim(f) = $n - 3$				
ε	$(\dots, 1, 1, 1)$	$(\dots, -1, 1, 1)$	$(\dots, 1, -1, 1)$	$(\dots, -1, -1, 1)$
C_ε	Card($f_1^{\uparrow+}$) ≥ 1	Card($f_1^{\uparrow+}$) ≥ 2	Card($f_1^{\uparrow+}$) ≥ 3	Card($f_1^{\uparrow+}$) ≥ 4 $f_1^{\uparrow+}$ not a trihedron
ε	$(\dots, 1, 1, -1)$	$(\dots, -1, 1, -1)$	$(\dots, 1, -1, -1)$	$(\dots, -1, -1, -1)$
C_ε	Card($f_1^{\uparrow+}$) ≥ 5 or $f_1^{\uparrow+}$ a trihedron	Card($f_1^{\uparrow+}$) ≥ 6	Card($f_1^{\uparrow+}$) ≥ 7	Card($f_1^{\uparrow+}$) = 8

From Proposition [5](#) and Definition [11](#) (needed when $f_1^{\uparrow+}$ is a trihedron), we derive the following corollary.

Corollary 1. *Let $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ be an ε -regular image. Let f be a k -face of \mathbb{F}^n (with $n - 3 \leq k \leq n - 1$). Then $\mu(f) = 1$ iff the set $f_1^{\uparrow+}$ satisfies the condition C_ε given in Table [2](#)*

3.4 Regular Images and Regular Open/Closed Sets

The object (resp. the background) of a regular image $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$, i.e., the set $\mu^{-1}(\{1\})$ (resp. $\mu^{-1}(\{0\})$) is topologically regular, i.e., it does not have thin parts nor thin holes (by “thin”, we mean of lower dimension than the surrounding space).

Proposition 6. *Let $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ be an ε -regular image. Let $x \in \{0, 1\}$. Then $(\mu^{-1}(\{x\}))^\circ$ is a regular open set and $(\mu^{-1}(\{x\}))^\downarrow$ is a regular closed set.*

Corollary 2. *Let $\mu : \mathbb{F}^n \rightarrow \{0, 1\}$ be an ε -regular image. If $\varepsilon = -1$, then $\mu^{-1}(\{1\})$ (resp. $\mu^{-1}(\{0\})$) is a regular open (resp. closed) set. If $\varepsilon = 1$, then $\mu^{-1}(\{1\})$ (resp. $\mu^{-1}(\{0\})$) is a regular closed (resp. open) set.*

4 Paths and (Digital) Fundamental Groups

In this section, we study how the functions ζ_ε behave relatively to the classical notions of path in \mathbb{Z}^n and \mathbb{F}^n . Theorem [2](#) states that ζ_ε induces a bijection between the set of the connected components of the object (resp. background) associated to an image $\lambda : \mathbb{Z}^n \rightarrow \{0, 1\}$, and the set of the connected components of the object (resp. background) associated to the regular image $\mu = \zeta_\varepsilon(\lambda)$, the function ε being chosen with respect to a given pair of adjacencies in \mathbb{Z}^n . Theorem [3](#) states that ζ_ε induces an isomorphism between the digital fundamental group of $\lambda^{-1}(\{1\})$ (resp. $\lambda^{-1}(\{0\})$) and the fundamental group of $\mu^{-1}(\{1\})$ (resp. $\mu^{-1}(\{0\})$).

4.1 Background Notions on Paths and Arcs

The fundamental group of topological spaces. Let X be a topological space. A path p in X is a continuous function $p : [0, 1] \rightarrow X$. Two paths p, q in X are *equivalent* if they have the same extremities (i.e., $p(0) = q(0)$ and $p(1) = q(1)$) and p can be continuously deformed to fit q , that is if there exists a continuous map $h : [0, 1] \times [0, 1] \rightarrow X$ such that, for all $t \in [0, 1]$, $h(t, 0) = p(t)$ and $h(t, 1) = q(t)$, and, for all $u \in [0, 1]$, $h(0, u) = p(0) = q(0)$ and $h(1, u) = p(1) = q(1)$ (the map h is called a *path-homotopy*). This relation on paths is actually an equivalence relation. We write $[p]$ for the equivalence class of p . If p, q are two paths in X such that $p(1) = q(0)$ we can define the product $p \cdot q$ by

$$(p \cdot q)(t) = \begin{cases} p(2t) & \text{if } t \in [0, \frac{1}{2}] \\ q(2t - 1) & \text{if } t \in [\frac{1}{2}, 1] \end{cases}$$

This product is well defined on equivalence classes by $[p] \cdot [q] = [p \cdot q]$. Let x be a point of X . A *loop* at x is a path in X which starts and ends at x . The product of two loops at x is a loop at x and the set $\pi(X, x)$ of equivalence classes of loops at x is a group for this product. It is called the *fundamental group* of X (with *basepoint* x).

Finite paths in posets. In a poset X , a function $f : [0, 1] \rightarrow X$ is a *step function* if there exist finitely many intervals $(I_i)_{i=0}^r$ ($r \geq 0$) such that f is constant on each interval I_i and $[0, 1] = \bigcup_{i=0}^r I_i$. A *finite path* in X is a path in X which is a step function. The sequence $(I_i)_{i=0}^r$ is called the *intervals sequence* of p and the sequence $(x_i)_{i=0}^r$ of the values of f on the track of p . A finite path is *regular* if there is no singleton in its intervals sequence.

The *product* of two arcs $(x_i)_{i=0}^r$ and $(y_i)_{i=0}^s$ is defined by $(x_i)_{i=0}^r \cdot (y_i)_{i=0}^s = (x_0, \dots, x_r, y_1, \dots, y_s)$ provided that $x_r = y_0$.

An arc $\chi = (x_i)_{i=0}^r$ ($r \geq 2$) is an *elementary stretching* (in X) of an arc χ' if for some $j \in \llbracket 1, r - 1 \rrbracket$, $\chi' = (x_i)_{i=0, i \neq j}^r$ or $(x_{j-1} = x_{j+1}$ and $\chi' = (x_i)_{i=0, i \neq j-1, i \neq j}^r$). An arc χ is a *deformation* (in X) of an arc χ' if there exists a sequence $(\chi_i)_{i=0}^s$ of arcs in X such that $\chi_0 = \chi, \chi_s = \chi'$ and for any $i \in \llbracket 1, s \rrbracket$, either χ_i is an elementary stretching of χ_{i-1} or χ_{i-1} is an elementary stretching in X of χ_i .

Let x be a point in X . “Being a deformation or equal” is an equivalence relation in the set of arcs in X from x to x . The set of equivalence classes, denoted by $\rho(X, x)$, is a group for the arc product.

Theorem 1 ([18]). *Let $x \in X$. The fundamental group $\pi(X, x)$ of X with basepoint x is isomorphic to the group $\rho(X, x)$.*

The digital fundamental group of \mathbb{Z}^n . A discrete analogue of the concept of fundamental group has been proposed in digital topology [10]. Let $n \in \{2, 3\}$ and $X \subseteq \mathbb{Z}^n$. The definition of the product for digital paths is straightforward but not so is the notion of equivalence between digital paths or loops. Two paths in X with same extremities are *directly equivalent* (in X) if they differ only in a unit lattice cube of \mathbb{Z}^n provided that, if $n = 3$ and the pair of adjacencies is $(6, 26)$, the cube must not contain two diametrically opposite points not in X . Finally, two paths in X , p_0, p_t ($t \geq 0$), with same extremities are *equivalent* (in X) if there is a sequence $(p_i)_{i=0}^t$ of paths in X such that, for all $i \in \llbracket 1, t \rrbracket$, p_i is directly equivalent in X to p_{i-1} .

4.2 Mapping Paths in \mathbb{Z}^n onto Arcs in \mathbb{F}^n

Let χ and χ' be two arcs in \mathbb{F}^n . We write $\chi \leq \chi'$ if there exist two paths $p \leq p'$ in \mathbb{F}^n whose tracks are χ and χ' (all paths in \mathbb{F}^n considered in the sequel are regular finite paths).

Definition 2. Let ω be an adjacency relation on \mathbb{Z}^n and $\gamma = (p_i)_{i=0}^r$ ($r \geq 0$) be an ω -path in \mathbb{Z}^n , given in its reduced form ($p_i \neq p_{i-1}$ for all $i \in \llbracket 1, r \rrbracket$). We define the arc $\zeta(\gamma)$ in \mathbb{F}^n by $\zeta(\gamma) = (q_j)_{j=0}^{2r}$ with $q_j = p_{\frac{j}{2}}$ if j is even and $q_j = q_{j-1} \cap q_{j+1}$ if j is odd, for all $j \in \llbracket 0, 2r \rrbracket$.

It is obvious that the sequence of faces $\zeta(\gamma)$ defined above is actually an arc in \mathbb{F}^n which is itself the track of a regular finite path in \mathbb{F}^n [18].

The following proposition states that ζ associates to a path in the object (resp. in the background), of a digital image λ , an arc in the object (resp. in the background) of the complex image $\zeta_\varepsilon(\lambda)$ under the condition that the connectivity function ε has been well chosen. The main consequence of this proposition is that the images of the connected components of the digital object (resp. background) are included in the connected components of the image of the object (resp. background).

Proposition 7. Let (α, β) be a pair of adjacencies on \mathbb{Z}^n . Let ε be the connectivity function associated to (α, β) . Let $x \in \{0, 1\}$. Let $\omega = \alpha$ if $x = 1$ and $\omega = \beta$ if $x = 0$. Let $\lambda : \mathbb{Z}^n \rightarrow \{0, 1\}$ be an image in \mathbb{Z}^n and $\mu = \zeta_\varepsilon(\lambda)$ be the corresponding image in \mathbb{F}^n . Let $\gamma = (p_i)_{i=0}^r$ ($r \geq 0$) be an ω -path in $\lambda^{-1}(\{x\})$. Then, $\zeta(\gamma)$ is an arc in $\mu^{-1}(\{x\})$.

The following proposition is straightforward.

Proposition 8. Let ω be an adjacency relation on \mathbb{Z}^n . The corresponding function ζ is a homomorphism for the paths product and the arc product: for all ω -paths $\gamma, \gamma' \in \mathbb{Z}^n$, $\zeta(\gamma \cdot \gamma') = \zeta(\gamma) \cdot \zeta(\gamma')$.

The injectivity of ζ is obvious since two distinct n -xels $a, b \in \mathbb{Z}^n$ are associated to distinct facets $\hat{a}, \hat{b} \in \mathbb{F}^n$. Proposition 9 establishes the surjectivity up to deformations: any arc χ from \hat{a} to \hat{b} in an object (resp. in the background) of the complex image is the deformation of an arc $\zeta(\gamma)$ for some path γ from a to b of the object (resp. background) of its associated digital image (if the complex image is associated to such a digital image). Proposition 9 is illustrated by Figure 6.

Proposition 9. Let (α, β) be a pair of adjacencies on \mathbb{Z}^n . Let ε be the connectivity function associated to (α, β) . Let $x \in \{0, 1\}$. Let $\omega = \alpha$ if $x = 1$ and $\omega = \beta$ if $x = 0$. Let $\lambda : \mathbb{Z}^n \rightarrow \{0, 1\}$ be an image in \mathbb{Z}^n and $\mu = \zeta_\varepsilon(\lambda)$ be the corresponding image in \mathbb{F}^n . Let $a, b \in \mathbb{Z}^n$. Let χ be an arc from the facet \hat{a} to the facet \hat{b} in $\mu^{-1}(\{x\})$. Then, there exists an ω -path γ from a to b in $\lambda^{-1}(\{x\})$ such that $\zeta(\gamma)$ is a deformation in $\mu^{-1}(\{x\})$ of χ .

Theorem 2. Let (α, β) be a pair of adjacencies in \mathbb{Z}^n . Let ε be the connectivity function associated to (α, β) . Let $\lambda : \mathbb{Z}^n \rightarrow \{0, 1\}$ be a digital image and $\zeta_\varepsilon(\lambda) = \mu : \mathbb{F}^n \rightarrow \{0, 1\}$ be the corresponding complex image. Let $x \in \{0, 1\}$. Then the function ζ_ε induces a one-to-one correspondence between the connected components of $\lambda^{-1}(\{x\})$ and the connected components of $\mu^{-1}(\{x\})$.

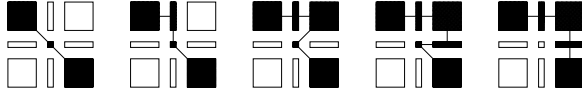


Fig. 6. From an arc μ to an arc $\zeta(\gamma)$ in a (-1) -regular image (the 9 points are in $\mu^{-1}(x)$)

4.3 Fundamental Groups

The aim of this section is to compare the digital fundamental group defined by Kong [10] for subsets of \mathbb{Z}^n , $n \in \{2, 3\}$, with the fundamental group of subspaces of \mathbb{F}^n . Thanks to Theorem 1 we can use arcs as well as paths in \mathbb{F}^n in order to perform this comparison.

Proposition 10. *Let (α, β) be a pair of adjacencies on \mathbb{Z}^n . Let ε be the connectivity function associated to (α, β) . Let $\lambda : \mathbb{Z}^n \rightarrow \{0, 1\}$ be a digital image and $\zeta_\varepsilon(\lambda) = \mu : \mathbb{F}^n \rightarrow \{0, 1\}$ be the corresponding complex image. Let $x \in \{0, 1\}$. Let $\omega = \alpha$ if $x = 1$ and $\omega = \beta$ if $x = 0$. Two ω -paths in $\lambda^{-1}(\{x\})$, γ and γ' , are equivalent in $\lambda^{-1}(\{x\})$ iff the arc $\zeta(\gamma)$ is equal to or is a deformation in $\mu^{-1}(\{x\})$ of the arc $\zeta(\gamma')$.*

Let $a \in \lambda^{-1}(\{x\})$. Let $\pi_D(\lambda^{-1}(\{x\}), a)$ be the digital fundamental group of $\lambda^{-1}(\{x\})$ with basepoint a and $\rho(\mu^{-1}(\{x\}), \dot{a})$ be the group of arcs in $\mu^{-1}(x)$ from \dot{a} to \dot{a} , up to deformations.

From Propositions 7 and 10, we know that the function $\check{\zeta}$ defined by

$$\left| \begin{array}{ccc} \check{\zeta} : & \pi_D(\lambda^{-1}(x), a) & \rightarrow \rho(\mu^{-1}(x), \dot{a}) \\ & [y] & \mapsto [\zeta(\gamma)] \end{array} \right.$$

where $[y]$ denotes the equivalence class of y (for the equivalence relation on digital paths of \mathbb{Z}^n on the left side and for the deformation on arcs of \mathbb{F}^n on the right side), is well-defined. Proposition 8 then states that $\check{\zeta}$ is a morphism. Propositions 9 and 10 give the surjectivity and the injectivity of $\check{\zeta}$, respectively. We conclude that the groups $\pi_D(\lambda^{-1}(\{x\}), a)$ and $\rho(\mu^{-1}(\{x\}), \dot{a})$ are isomorphic and, since $\rho(\mu^{-1}(\{x\}), \dot{a})$ and $\pi(\mu^{-1}(\{x\}), \dot{a})$ are isomorphic (Theorem 1), the following theorem holds.

Theorem 3. *Let (α, β) be a pair of adjacencies on \mathbb{Z}^n . Let ε be the connectivity function associated to (α, β) . Let $\lambda : \mathbb{Z}^n \rightarrow \{0, 1\}$ be an image in \mathbb{Z}^n and $\mu = \zeta_\varepsilon(\lambda)$ be the corresponding image in \mathbb{F}^n . For any $a \in \lambda^{-1}(\{x\})$, the digital fundamental group of $\lambda^{-1}(\{x\})$ with basepoint a is isomorphic to the fundamental group of the poset $(\mu^{-1}(\{x\}), \subseteq)$ with base point \dot{a} .*

5 Conclusion

In this article, a *modus operandi* has been proposed to embed digital binary images, equipped with a pair of standard adjacencies, in the space of cubical complexes. In particular, it has been proved that it preserves the connected components of both object and background and, more generally, preserves also the (digital) fundamental groups.

These results, associated to those of [18], justify the soundness of all the contributions previously devoted to design homotopy type-preserving binary image processing

methods, especially concerning the correctness of their behaviour with respect to the “continuous” topology of the handled digital objects. They also permit to establish links between image processing methods developed either in classical digital spaces (\mathbb{Z}^n) or cubical complexes (\mathbb{F}^n), and to potentially unify some of them. An extended version of this work (with proofs of the propositions and theorems) will be proposed soon [17].

References

1. Aktouf, Z., Bertrand, G., Perrotton, L.: A three-dimensional holes closing algorithm. *Pattern Recognition Letters* 23(5), 523–531 (2002)
2. Ayala, R., Domínguez, E., Francés, A.R., Quintero, A.: Digital lighting functions. In: Ahronovitz, E. (ed.) *DGCI 1997. LNCS*, vol. 1347, pp. 139–150. Springer, Heidelberg (1997)
3. Ayala, R., Domínguez, E., Francés, A.R., Quintero, A.: Digital homotopy with obstacles. *Discrete Applied Mathematics* 139(1–3), 5–30 (2004)
4. Bertrand, G.: New notions for discrete topology. In: Bertrand, G., Couprie, M., Perrotton, L. (eds.) *DGCI 1999. LNCS*, vol. 1568, pp. 218–228. Springer, Heidelberg (1999)
5. Bertrand, G., Couprie, M.: A model for digital topology. In: Bertrand, G., Couprie, M., Perrotton, L. (eds.) *DGCI 1999. LNCS*, vol. 1568, pp. 229–241. Springer, Heidelberg (1999)
6. Bertrand, G., Malandain, G.: A new characterization of three-dimensional simple points. *Pattern Recognition Letters* 15(2), 169–175 (1994)
7. Faisan, S., Passat, N., Noblet, V., Chabrier, R., Meyer, C.: Topology-preserving warping of binary images according to one-to-one mappings. *IEEE Transactions on Image Processing* (to appear)
8. Gonzalez-Diaz, R., Jimenez, M.J., Medrano, B.: Well-composed cell complexes (November 2010), <http://personal.us.es/majiro/ctic19.pdf>, communication at CTIC 2010
9. Khalimsky, E., Kopperman, R., Meyer, P.R.: Computer graphics and connected topologies on finite ordered sets. *Topology and its Applications* 36(1), 1–17 (1990)
10. Kong, T.Y.: A digital fundamental group. *Computers and Graphics* 13(2), 159–166 (1989)
11. Kong, T.Y., Roscoe, A.W.: A theory of binary digital pictures. *Computer Vision, Graphics and Image Processing* 32(2), 221–243 (1985)
12. Kong, T.Y.: Topology-preserving deletion of 1’s from 2-, 3- and 4-dimensional binary images. In: Ahronovitz, E. (ed.) *DGCI 1997. LNCS*, vol. 1347, pp. 3–18. Springer, Heidelberg (1997)
13. Kong, T.Y., Rosenfeld, A.: Digital topology: introduction and survey. *Computer Vision, Graphics and Image Processing* 48(3), 357–393 (1989)
14. Kovalevsky, V.A.: Finite topology as applied to image analysis. *Computer Vision, Graphics, and Image Processing* 46(2), 141–161 (1989)
15. Lachaud, J.O., Montanvert, A.: Continuous analogs of digital boundaries: A topological approach to iso-surfaces. *Graphical Models and Image Processing* 62, 129–164 (2000)
16. Latecki, L.J.: 3d well-composed pictures. *Graph. Models Image Process.* 59(3), 164–172 (1997)
17. Mazo, L., Passat, N., Couprie, M., Ronse, C.: Digital imaging: A unified topological framework. Tech. Rep. hal-00512270, Université Paris-Est (2010), <http://hal.archives-ouvertes.fr/hal-00512270/fr/>
18. Mazo, L., Passat, N., Couprie, M., Ronse, C.: Paths, homotopy and reduction in digital images. *Acta Applicandae Mathematicae* 113(2), 167–193 (2011)
19. Rosenfeld, A.: Connectivity in digital pictures. *Journal of the Association for Computer Machinery* 17(1), 146–160 (1970)

Isthmus-Based 6-Directional Parallel Thinning Algorithms

Benjamin Raynal and Michel Couprie

Université Paris-Est
Laboratoire d'Informatique Gaspard Monge, Equipe A3SI

Abstract. Skeletons are widely used in pattern recognition and image analysis. A way to obtain skeletons is the thinning approach, consisting in iteratively removing points from the object without changing the topology. In order to preserve geometric information, it is usual to preserve curve end points (for curve skeletons) or surface end points (for surface skeletons). In this paper we propose a new fast directional parallel thinning scheme, preserving isthmuses (a generalization of curve/surface interior points), and providing skeletons with low amount of noise. We also prove the topology preservation of our approach.

Keywords: Thinning Algorithm, Surface Skeleton, Curvilinear Skeleton, Topology Preservation, Directional Thinning, Isthmus.

1 Introduction

Skeletons were originally defined by Blum [1] based on a “grass fire” analogy. Imagine a shape as a field covered by dry grass; if you set on fire the contour of the field, then the meeting points of the flame fronts would constitute the skeleton of the shape.

Various methods can be used to obtain skeletons: Voronoi-based transformations [2,3], distance-based transformations [4,5], general-field methods [6,7] and thinning.

Thinning consists of iteratively removing points of the object (called *simple points*) without changing its topology, until stability. Without constraints, a thinning process results in a *ultimate skeleton*, which has the same topology as the original object, and no simple point. However, ultimate skeletons are not efficient as shape descriptors, due to the fact that they may fail to preserve important geometric information. In 3D, two other kinds of skeletons can be obtained, which are closer to the original Blum’s analogy: *curvilinear skeletons* (1D) and *surface skeletons* (2D). Figure 1 shows examples of different skeletons. In the literature, in order to obtain such skeletons, it is usual to constrain the thinning to preserve the skeleton extremity points (i.e. curve end points or surface end points).

A common problem encountered in applications is linked to the sensibility of the skeletonization process to small irregularities of the contour: the skeleton may contain “spurious branches” that would make it difficult to exploit.



Fig. 1. From the left to the right (in red):ultimate, curve and surface skeletons

Thinning algorithms have usually either low computational cost, or good quality results (i.e. few spurious branches), but rarely both. In this paper, we propose a new fast parallel directional thinning algorithm constrained by 1D-isthmuses (a generalization of curve interior points) instead of curve end points, resulting in curve skeletons with very few spurious branches. We also propose variations to provide surface skeletons and ultimate skeletons.

This paper is organized as follows: In the sequel of this section, we recall basic notions of digital topology necessary to understand thinning theory, then we present the different strategies of thinning. In Sect. 2, we present the works on which our method is based. In Sect. 3, we introduce our new curve-thinning algorithm, and prove the preservation of topology. We also introduce variations for surface skeletons and ultimate skeletons. Finally, in Sect. 4, we show some results of our algorithms and compare them to those of classical algorithms of the same kind.

1.1 Basic Notions of Digital Topology

In 3D digital topology, the framework is the *discrete grid* \mathbb{Z}^3 . The *object* is represented by $X \subset \mathbb{Z}^3$, and its *complementary* $\mathbb{Z}^3 \setminus X$ is denoted by \bar{X} . A *point* $p \in \mathbb{Z}^3$ is defined by a triplet of integers (p_1, p_2, p_3) .

The notion of *neighborhood* is central for digital topology. In 3D, several neighborhoods are considered: the *6-neighborhood* of p is the set $N_6(p) = \{q \in \mathbb{Z}^3; |q_1 - p_1| + |q_2 - p_2| + |q_3 - p_3| \leq 1\}$; the *26-neighborhood* of p is the set $N_{26}(p) = \{q \in \mathbb{Z}^3; \max(|q_1 - p_1|, |q_2 - p_2|, |q_3 - p_3|) \leq 1\}$. For a k -neighborhood, we define $N_k^*(p) = N_k(p) \setminus \{p\}$.

Based on the notion of neighborhood we can define the notion of connectivity. Let p be an element of X , we define the *k -connected component of X containing p* , denoted by $C_k(p, X)$, as the maximal subset of X containing p , such that for all points $a \in C_k(p, X)$, there exists a sequence of points of X $\langle p_0, \dots, p_n \rangle$ such that:

- $p_0 = p$ and $p_n = a$,
- $\forall i \in \{1, \dots, n\}, p_{i-1}$ is in the k -neighborhood of p_i .

The set of all k -connected components of X is denoted by $C_k(X)$. A subset Y of \mathbb{Z}^3 is *k -adjacent* to a point $p \in \mathbb{Z}^3$ if $Y \cap N_k^*(p) \neq \emptyset$. The set of all k -connected

components of X which are k -adjacent to a point p is denoted by $C_k^p(X)$. When we consider the k -connectivity of X , it is mandatory to consider a different \bar{k} -connectivity for \bar{X} [8]. In 3D, if $k = 6$, then $\bar{k} = 26$ and inversely. This rule is necessary in order to retrieve some important topological properties such as the Jordan theorem.

Connectivity Numbers. In order to provide a local description of simple points and other characteristic points, Bertrand and Malandain [9] introduced the notion of connectivity numbers. In 3D, the definition of connectivity numbers lies on the notion of *geodesic neighborhood*. Let $X \subseteq \mathbb{Z}^3$ and $p \in \mathbb{Z}^3$. The t -order k -geodesic neighborhood of p in X is the set $N_k^t(p, X)$ recursively defined by:

- $N_k^1(p, X) = N_k^*(p) \cap X$
- $N_k^t(p, X) = \bigcup \{N_k(q) \cap N_{26}^*(p) \cap X, q \in N_k^{t-1}(p, X)\}$

The geodesic neighborhoods $G_k(p, X)$ are defined by: $G_6(p, X) = N_6^2(p, X)$ and $G_{26}(p, X) = N_{26}^1(p, X)$. We can now define the connectivity numbers in 3D, for $k \in \{6, 26\}$ as: $T_k(p, X) = |C_k(G_k(p, X))|$, where $|S|$ denotes the number of elements of the set S .

Simple points. Intuitively, a point is simple for X if it can be removed from X without changing its topology. In the digital topology framework, the topology of an object depends on the chosen connectivity; for this reason, when considering a k -connected object, we will talk about k -simple points. The notion of simple point is central for homotopic thinning in the digital framework: a skeleton is obtained by removing iteratively simple points from an object. In 3D, the removal of a point may not only change the number of connected components of the object or its complementary, but may also change the tunnels of the object. Bertrand and Malandain propose a local characterization of simple points using connectivity numbers T_6 and T_{26} :

Proposition 1. [9] *Let $X \subseteq \mathbb{Z}^3$ and $x \in X$. The point x is k -simple for X iff $T_k(x, X) = 1$ and $T_{\bar{k}}(x, \bar{X}) = 1$.*

1.2 Thinning Algorithms

Thinning in the digital framework consists of removing simple points from an object, until either no more simple point can be found (resulting in a ultimate skeleton), or a satisfactory subset of voxels has been reached (resulting in a curvilinear of surface skeleton). Two main strategies are possible for removing simple points: sequential removal and parallel removal.

Sequential algorithms. Sequential removal of simple points can be achieved by detecting simple points in an object, and removing them one after the other, until no more simple point can be found. After removing a simple point, the new set of simple points of the object must be computed. Such basic strategy does not guarantee the result, which is an ultimate skeleton, to be centered in the original

object. It is important, when designing a sequential thinning algorithm, to decide of a removal order of simple points, and of a strategy for preserving interesting visual features of the object. A widely used strategy to obtain a centered skeleton with a sequential thinning process consists of computing a priority function (e.g. distance to the background) on the object and removing the simple points of X according to the value of this function [10] : at each step, the simple point that is removed is one with the lowest possible value.

Parallel algorithms. Whereas iterative algorithms remove only one simple point at a time, parallel algorithms consists in removing a set of simple points simultaneously. The main problem of parallel thinning is that removing simple points simultaneously from an object usually "breaks" the topology. Thus, additional conditions must be introduced in order to solve this problem.

For a given algorithm, the *deletion conditions* are the conditions that a point has to satisfy to be removable. The *support* is the minimal set of points whose values must be examined to verify the deletion conditions for a point $p \in X$. According to Hall [11], parallel thinning algorithms can be divided into three categories:

- In *directional algorithms*, the main loop is divided into sub-iterations, and the deletion conditions are changed from one sub-iteration to another. The most usual kind of directional algorithm use 6 sub-iterations [12,13,15,14]. Directional algorithms with 3 sub-iterations [16], 8 sub-iterations [17] and 12 sub-iterations [18] have also been proposed.
- In *subfield-based algorithms*, the points of the object are decomposed into subfields, and at a given iteration of the algorithm, only points in a given subfield are studied. Algorithms using 2 subfields [19], 4 subfields [20] and 8 subfields [20] have been proposed.
- In *fully parallel algorithms*, no sub-iteration takes place : the same thinning operator is used at each iteration of the main loop [21,22,23,24].

Parallel algorithms are generally more robust to contour noise than sequential ones. Subfield-based algorithms usually produce shaky skeletons, and fully parallel algorithms require an extended support (usually, $5 \times 5 \times 5$), while for other strategies the usual support is a $3 \times 3 \times 3$ neighborhood (i.e. N_{26}).

For our aims, a parallel directional algorithm is the best choice.

2 Background

Our method is based on two different works. We use constraints that were originally introduced by Bertrand and Couprie [25] in the context of sequential thinning, and the design of the masks used in our parallel directional algorithm is based on those defined by Palágyi and Kuba [12].

2.1 Bertrand and Couprie Isthmus Based Thinning

In 2007, Bertrand and Couprie [25] proposed a sequential thinning based on isthmuses.

Definition 1. Let $X \subseteq \mathbb{Z}^3, p \in X$,

- p is a 1D-isthmus iff $T_k(p, X) \geq 2$.
- p is a 2D-isthmus iff $T_{\bar{k}}(p, \bar{X}) \geq 2$.

The strategy used by Bertrand and Couprie consists in dynamically detecting the considered isthmuses (1D or 2D) and accumulating them in a constraint set. A point can be deleted if it is simple and not in the constraint set. By this way, a point detected as an isthmus in a given iteration cannot be removed later. Depending on the considered isthmuses, the method provides curvilinear skeletons (in case of 1D isthmuses) or surface skeletons (in case of 2D isthmuses).

The consideration of isthmuses instead of extremities points is interesting for two reasons: isthmuses can be easily locally defined and detected (it is not the case of surface end points), and appear less often than extremities during thinning process, leading to less noisy skeletons. However, the sequential approach is less adapted than parallel ones for this purpose, and provides skeletons of lower quality.

2.2 Palágyi and Kuba 6-Directional Thinning

In 1998, Palágyi and Kuba [12] proposed a 6-directional thinning algorithm producing curvilinear skeletons, which we denote by PKD6.

Each iteration of PKD6 consists of 6 sub-iterations in which points are removed from the shape if and only if the configuration of their 26-neighborhood matches at least one mask of the mask set \mathcal{M}_d for the given direction d . The mask set \mathcal{M}_U used for the *UP* direction is presented in Fig. 2. Mask sets $\mathcal{M}_D, \mathcal{M}_N, \mathcal{M}_S, \mathcal{M}_E, \mathcal{M}_W$ for the other directions (*DOWN*, *NORTH*, *SOUTH*, *EAST* and *WEST*) are obtained by appropriate rotations and reflections. The algorithm is summarized in Alg. 1. Palágyi and Kuba proved that their algo-

Algorithm 1. PKD6 [12]

Data: $X \subseteq \mathbb{Z}^3$
Result: A skeleton of X

- 1 $Y = X$
- 2 **repeat**
- 3 **foreach** *direction* d in (U, D, N, S, E, W) **do**
- 4 $Y = Y \setminus \{p \in Y; p \text{ matching one mask of } \mathcal{M}_d\}$
- 5 **until** *stability*;
- 6 **return** Y

rithm preserves topology, using the following theorem (Theorem 1) proved by Ma in [26].

Definition 2. [26] Let $X \subseteq \mathbb{Z}^3$ be an object. The set $D = \{d_1, \dots, d_k\} \subseteq X$ is called a simple set of X if D can be arranged in a sequence $\langle d_{i_1}, \dots, d_{i_k} \rangle$ in which each d_{i_j} is simple for $X \setminus \{d_{i_1}, \dots, d_{i_{j-1}}\}$ for $j = 1, \dots, k$.

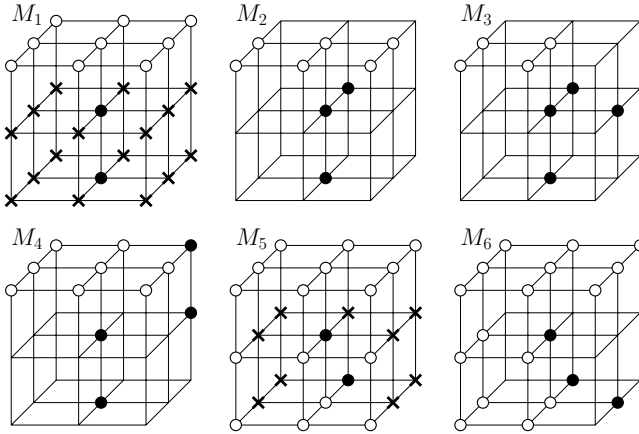


Fig. 2. Base masks M_1 - M_6 and their rotations around the vertical axis form the set of masks \mathcal{M}_U for direction UP . Points marked by a black disk have to belong to the object, points marked by a white disk have to be out of the object, and at least one point marked by a cross in masks M_1 and M_5 has to belong to the object.

Definition 3. [26] A 3D parallel reduction operation is said to preserve topology if, for all possible 3D pictures, the set of all points that are deleted simultaneously is simple.

Theorem 1. [26] A 3D parallel reduction operation preserves topology if all of the following conditions hold:

1. Only simple points are deleted.
2. If two points belonging to the object, p and q , of a 2×2 square in \mathbb{Z}^3 are deleted, then the set $\{p, q\}$ is simple.
3. If three points belonging to the object, p, q and r , of a 2×2 square in \mathbb{Z}^3 are deleted, then the set $\{p, q, r\}$ is simple.
4. If four points belonging to the object, p, q, r and s , of a 2×2 square in \mathbb{Z}^3 are deleted, then the set $\{p, q, r, s\}$ is simple.
5. No component of the object contained in a $2 \times 2 \times 2$ cube in \mathbb{Z}^3 can be deleted completely.

Theorem 2. [12] The thinning algorithm PKD6 preserves topology.

3 New Method

We propose a 6-directional thinning algorithm based on isthmuses. Our algorithm is designed in two steps, separating the thinning process (using a 6-directional approach) and the isthmuses detection. Each iteration consists of:

1. updating the constraint set, by adding points of the object detected as isthmuses,

Algorithm 2. D6IID

Data: $X \subseteq \mathbb{Z}^3$, $K \subseteq X$
Result: A skeleton of X

```

1 repeat
2    $K = K \cup \Psi_{1D}(X)$ 
3   foreach direction  $d$  in  $(U, D, N, S, E, W)$  do
4      $X = \tau_d(X, K)$ 
5 until stability;
6 return  $X$ 

```

2. removing deletable points of the object which are not in the constraint set (see Algo 2).

This design allows for the preservation of some points defined a priori by the initial set K (which may be empty). The function $\Psi_{1D}(X)$ returns the set of all the 1D-isthmuses of X (see Def. 1). The function $\tau_d(X, K)$ returns the set $X \setminus S$, S being the set of all points from $X \setminus K$ respecting deletion conditions in X for direction d . This function is defined in the following of this section.

3.1 Design of Deletion Condition Masks

Contrary to deletion conditions used in directional algorithms found in the literature, those of τ_d permit the deletion of curve end points, the constraint points being considered separately.

Deletion conditions can be represented using a set of masks: a point respects deletion conditions if and only if it matches at least one of the masks. In this section, we present the masks used by τ_d , obtained by modifying those used by PKD6.

In order to separate geometrical and topological conditions, we have to allow in τ_d the deletion of ending points (i.e. points with only one neighbor in the object). For this purpose, we add three masks to \mathcal{M}_d , representing ending points which may be removed for the direction d . These masks are shown in Fig. 3 for the case $d = U$.

Proving that our algorithms preserve topology (Prop 2 and Prop 3) can be done using the framework of critical kernels, introduced by Bertrand [27]. This framework is indeed the most powerful one for both proving and designing n -dimensional homotopic thinning algorithms. However, in this particular case it is simpler to build on the proof that was given by Palágyi and Kuba for PKD6, as our algorithm only slightly differs from the latter.

Proposition 2. *Algorithm PKD6 with mask sets \mathcal{M}_d extended by adding convenient rotations of M_α , M_β and M_γ preserves topology.*

Proof. We prove that all conditions of theorem 1 hold.

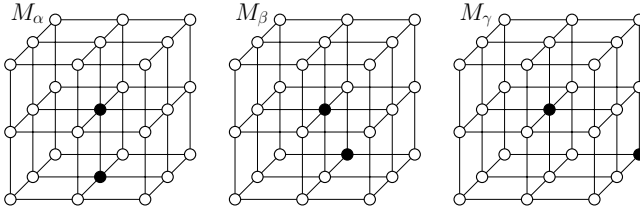


Fig. 3. Masks M_α - M_γ and their rotations around vertical axis are added to the set of masks \mathcal{M}_U for direction UP in order to remove ending points

Let $X \subseteq \mathbb{Z}^3$ and $p \in X$. If p matches a mask in \mathcal{M}_U , then we know that it is simple from theorem 2. If p matches a mask in $\{M_\alpha, M_\beta, M_\gamma\}$, then p is obviously simple.

Let Z be a set of object points contained in a 2×2 square in \mathbb{Z}^3 , or being an object component contained in a $2 \times 2 \times 2$ cube in \mathbb{Z}^3 , with $|Z| \geq 2$.

If all points of Z match masks of \mathcal{M}_U , then we know that all conditions of theorem 1 hold, from theorem 2.

Otherwise, if one point p of Z matches a mask in $\{M_\alpha, M_\beta, M_\gamma\}$, then clearly $|Z| = 2$. Let q be the other point in Z . By examination of all configurations, it can be seen that any mask in $\mathcal{M}_U \cup \{M_\alpha, M_\beta, M_\gamma\}$ positioned on q does not match, thus Z cannot be deleted. \square

3.2 Mask Set Reduction

The set of masks $\mathcal{M}_U \cup \{M_\alpha, M_\beta, M_\gamma\}$ can be compacted as proposed in Fig 4. We can observe that mask M'_1 matches exactly the same configurations as those matched by masks M_1 and M_α . In the same way, mask M'_5 matches exactly the same configurations as those matched by masks M_5 and M_β . Masks M'_2, M'_3, M'_4, M'_6 and M'_7 are respectively the same as M_2, M_3, M_4, M_6 and M_γ .

Using the sets \mathcal{M}'_d with $d \in \{U, D, N, S, E, W\}$, we can now define τ_d , as proposed in algorithm 3.

Algorithm 3. τ_d

Data: $X \subseteq \mathbb{Z}^3, K \subseteq X$

Result: A subset of X

- 1 $R =$ set of points matching a mask belonging to \mathcal{M}'_d
 - 2 $R = R \setminus K$
 - 3 **return** $X \setminus R$
-

Proposition 3. *Algorithm D6I1D preserves topology.*

Proof. From proposition 2, algorithm PKD6 with mask sets \mathcal{M}_d extended by adding convenient rotations of M_α, M_β and M_γ preserves topology. We observe that:

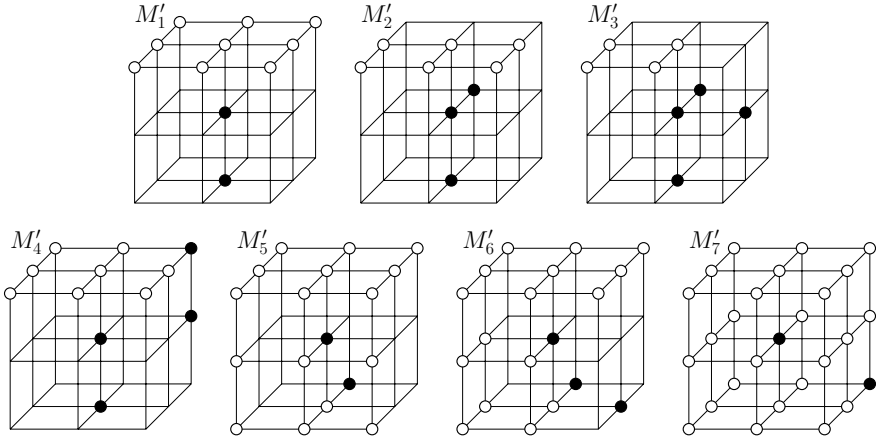


Fig. 4. Final set of masks \mathcal{M}'_U for direction UP

- D6I1D differs only from this algorithm by constraining some points to be preserved;
- if theorem [11](#) holds for an operation A , then it also holds for an operation B that removes only a subset of the points removed by A .

These observations lead to the conclusion that D6I1D preserves topology. \square

3.3 Other Kinds of Skeletons

In fact, our algorithm offers great flexibility, due to the fact that constraint set detection (function Ψ) can be changed to implement other conditions. We provide two examples of such variations.

Ultimate Skeletons. In order to obtain ultimate skeletons instead of curvilinear ones, we propose a very simple variation of D6I1D, consisting in replacing Ψ_{I1D} by the function returning empty set (K is unchanged). We call this variation D6U.

Surface Skeletons. In order to obtain surface skeletons instead of curvilinear ones, we propose a very simple variation of D6I1D, consisting in replacing Ψ_{I1D} by a function Ψ_{I2D} returning the set of 2D-isthmuses (see Def. [11](#)), which can be locally detected. We call this variation D6I2D.

4 Comparative Results

We implemented our method in C++, using two usual optimizations: border lists [\[24\]](#), in order to check only border points in each iteration, and look-up

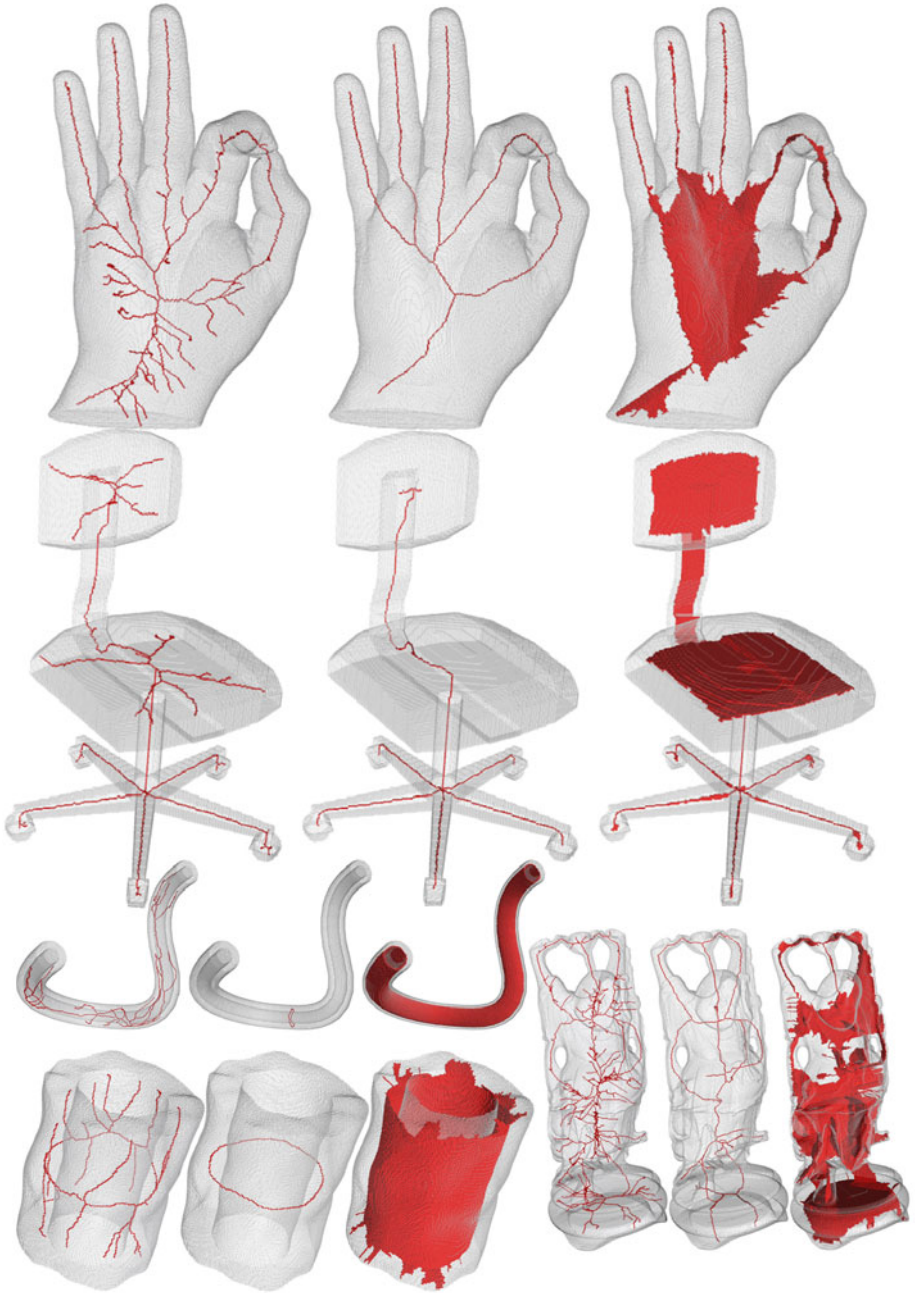


Fig. 5. Skeletons obtained with different algorithms. For each object, from the left to the right: PKD6, D6I1D, D6I2D. The used objects, from the left to the right, then from the top to the bottom: *hand*, *chair*, *pipe*, *cylinder* and *Buddha*.

Table 1. For each object, the size of the image, the numbers of object points (**OP**), and for both *D6I1D* and *D6I2D*, the number of skeleton points (**SP**), the number of iterations (**It**) and the computation time in seconds (**CT(s)**)

Object	Size	OP	PKD6			D6I1D			D6I2D		
			SP	It	CT(s)	SP	It	CT(s)	SP	It	CT(s)
<i>Buddha</i>	180 × 400 × 180	2 514 223	1 953	90	1.06	1 636	90	1.06	66 492	56	0.98
<i>chair</i>	300 × 215 × 215	907 601	855	81	0.35	814	82	0.35	20 238	28	0.33
<i>pipe</i>	300 × 300 × 300	663 586	224	386	1.00	78	385	1.03	54 562	10	0.36
<i>cylinder</i>	200 × 200 × 200	1 077 425	455	112	0.56	229	112	0.57	31 452	26	0.47
<i>hand</i>	185 × 350 × 220	2 281 776	899	94	0.80	863	94	0.81	13 756	55	0.79

tables in order to use pre-computation, for each possible configuration of 26-neighborhood, of the deletion tests and isthmuses detection. The results, obtained on a PC under Linux, with a processor Intel(R) Core(TM) 2 Quad Q8200 at 2.33 GHz, are presented in Table 1.

Concerning the quality of the resulting skeleton, as no general quality measurement exists (to our knowledge), we propose to visually compare the results of PKD6 and D6I1D (see Fig. 5). It can be observed that our algorithm provides results with less spurious branches than PKD6.

5 Conclusion

In this paper, we have proposed a new scheme based on isthmus, highly flexible, fast to compute, and providing different kinds of good quality skeletons.

References

1. Blum, H.: An associative machine for dealing with the visual field and some of its biological implications. *Biological Prototypes and Synthetic Systems* 1, 244–260 (1962)
2. Brandt, J.W., Algazi, V.R.: Continuous skeleton computation by voronoi diagram. *CVGIP: Image Understanding* 55(3), 329–338 (1992)
3. Näf, M., Székely, G., Kikinis, R., Shenton, M.E., Kübler, O.: 3d voronoi skeletons and their usage for the characterization and recognition of 3d organ shape. *CVGIP: Image Understanding* 66(2), 147–161 (1997)
4. Borgfors, G., Nyström, I., Sanniti Di Baja, G.: Computing skeletons in three dimensions. *Pattern Recognition* 32(7), 1225–1236 (1999)
5. Toriwaki, J.I., Mori, K.: Distance transformation and skeletonization of 3d pictures and their applications to medical images. In: *Digital and Image Geometry, Advanced Lectures*, pp. 412–428. Springer, London (2001)
6. Ahuja, N., Chuang, J.-H.: Shape representation using a generalized potential field model. *IEEE Trans. Pattern Anal. Mach. Intell.* 19(2), 169–176 (1997)
7. Rumpf, M., Telea, A.: A continuous skeletonization method based on level sets. In: *VISSYM 2002*, p. 151. Eurographics Association, Aire-la-Ville (2002)
8. Kong, T.Y., Rosenfeld, A.: Digital topology: introduction and survey. *Comp. Vision, Graphics and Image Proc.* 48, 357–393 (1989)

9. Bertrand, G., Malandain, G.: A new characterization of three-dimensional simple points. *Pattern Recognition Letters* 15(2), 169–175 (1994)
10. Davies, E.R., Plummer, A.P.N.: Thinning algorithms: A critique and a new methodology. *Pattern Recognition* 14(1-6), 53–63 (1981)
11. Hall, R.W.: Parallel connectivity-preserving thinning algorithms. *Machine Intelligence and Pattern Recognition* 19, 145–179 (1996)
12. Palágyi, K., Kuba, A.: A 3D 6-subiteration thinning algorithm for extracting medial lines. *Pattern Recognition Letters* 19(7), 613–627 (1998)
13. Lohou, C., Bertrand, G.: A 3d 6-subiteration curve thinning algorithm based on p-simple points. *Discrete Appl. Math.* 151(1-3), 198–228 (2005)
14. Gong, W., Bertrand, G.: A simple parallel 3D thinning algorithm. In: 10th International Conference on Pattern Recognition 1990 Proceedings, pp. 188–190 (1990)
15. Ma, C.M., Wan, S.Y.: Parallel thinning algorithms on 3D (18, 6) binary images. *Computer Vision and Image Understanding* 80(3), 364–378 (2000)
16. Palágyi, K.: A 3-Subiteration Surface-Thinning Algorithm. In: *Computer Analysis of Images and Patterns*, pp. 628–635. Springer, Heidelberg (2007)
17. Palágyi, K., Kuba, A.: Directional 3D thinning using 8 subiterations. In: *Discrete Geometry for Computer Imagery*, pp. 325–336. Springer, Heidelberg (1999)
18. Lohou, C., Bertrand, G.: A 3d 12-subiteration thinning algorithm based on p-simple points. *Discrete Appl. Math.* 139(1-3), 171–195 (2004)
19. Kardos, P., Németh, G., Palágyi, K.: Topology preserving 2-subfield 3d thinning algorithms. In: *Proceedings of the IASTED International Conference on Signal Processing, Pattern Recognition and Applications*, pp. 310–316. IASTED, Innsbruck (2010)
20. Németh, G., Kardos, P., Palágyi, K.: Topology Preserving 3D Thinning Algorithms Using Four and Eight Subfields. *Image Analysis and Recognition*, 316–325 (2010)
21. Manzanera, A., Bernard, T.M., Preteux, F., Longuet, B.: Medial faces from a concise 3D thinning algorithm. In: *Proc. 7th IEEE Internat. Conf. Computer Vision, ICCV 1999*, vol. 1, pp. 337–343 (1999)
22. Wang, T., Basu, A.: A note on 'A fully parallel 3D thinning algorithm and its applications'. *Pattern Recognition Letters* 28(4), 501–506 (2007)
23. Bertrand, G., Couprie, M.: A new 3D parallel thinning scheme based on critical kernels. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) *DGCI 2006. LNCS*, vol. 4245, pp. 580–591. Springer, Heidelberg (2006)
24. Palágyi, K.: A 3D fully parallel surface-thinning algorithm. *Theoretical Computer Science* 406(1-2), 119–135 (2008)
25. Bertrand, G., Couprie, M.: Transformations topologiques discrètes. In: Coeurjolly, D., Montanvert, A., Chassery, J. (eds.) *Géométrie discrète et images numériques*, Hermès, pp. 187–209 (2007)
26. Ma, C.M.: On topology preservation in 3d thinning. *CVGIP: Image Understanding* 59(3), 328–339 (1994)
27. Bertrand, G.: On critical kernels. *Comptes Rendus de l'Académie des Sciences, Série Math.* 1(345), 363–367 (2007)

Quasi-Linear Transformations, Numeration Systems and Fractals*

Marie-Andrée Jacob-Da Col and Pierre Tellier

LSIIT-UMR 7005, Pôle API, Bd Sébastien Brant
67412 Illkirch Cedex France
{dacolm,tellier}@unistra.fr

Abstract. In this paper we will define relations between quasi-linear transformations, numeration systems and fractals. A Quasi-Linear Transformation (QLT) is a transformation on \mathbb{Z}^n which corresponds to the composition of a linear transformation with an integer part function. We will first give some theoretical results about QLTs. We will then point out relations between QLTs, numeration systems and fractals. These relations allow us to define new numeration systems, fractals associated with them and n-dimensional fractals. With help of some properties of the QLTs we can give the fractal dimension of these fractals.

Keywords: Gaussian integers, algebraic integers, numeration systems, discrete linear transformations, fractals.

1 Introduction

Fractal tiles generated by numeration systems and substitutions have been widely studied, see for example [1], [2], [5]. In this paper we define discrete linear transformations called QLT which generate tilings of \mathbb{Z}^n . The tilings studied here are self-similar, they allow us to generate n-dimensional fractals. In dimension two we point out relations between QLTs and numeration systems which allow us to define new numeration systems. These discrete linear transformations are also in relation with discrete lines [17], [13].

Let g be a linear transformation from \mathbb{Z}^n to \mathbb{Q}^n , defined by a matrix $\frac{1}{w}A$ where A is an integer matrix and w a positive integer. If we compose this transformation with an integer part function we obtain a Quasi-Linear Transformation (QLT) from \mathbb{Z}^n to \mathbb{Z}^n . We will only consider the integer part function with a positive remainder. We will denote it $\lfloor \cdot \rfloor$. If x and y are two integers, $\lfloor \frac{x}{y} \rfloor$ denotes the quotient of the Euclidean division of x by y . We denote G as the QLT defined by g . Main research results on QLTs are listed in the first section of this paper.

Let us consider β an algebraic number and \mathcal{D} a set of elements of $\mathbb{Z}[\beta]$. β is a valid base using the digit set \mathcal{D} if every integer $c \in \mathbb{Z}[\beta]$ has a unique representation (decomposition) of the form: $c = \sum_{j=0}^n a_j \beta^j$, where $a_j \in \mathcal{D}$ and

* This work was supported by the Agence Nationale de la Recherche through contract ANR-2010-BLAN-0205-01.

$n \in \mathbb{N}$. Then (β, \mathcal{D}) is called a numeration system. The third section will point out some relations between QLTs and numeration systems when β is a Gaussian integer ($\beta = a+ib$, with $a, b \in \mathbb{Z}$) or an algebraic integer of order 2 ($\beta^2+b\beta+a = 0$ with $a, b \in \mathbb{Z}$). These relations and some properties of QLTs will allow us to generate new numeration systems.

In the fourth section we will see how QLTs generate fractals (some of them are in relation with numeration systems). In 2D the fractal dimension of the border of these fractals can be determined by substitutions associated with the QLT. These substitutions have been defined in [8] and [11]. For the n-dimensional fractals the fractal dimension is determined directly using properties of QLTs.

2 Quasi-Linear Transformations

In this section we recall definitions and results that can be found in [7], [10], [3], [4] and that are useful in the rest of the paper. The proposition [1] can be found in [10]. Proposition [2] extends to \mathbb{Z}^n a result given in [7] and [10] in \mathbb{Z}^2 .

Definition 1. Let g be a linear transformation from \mathbb{Z}^n to \mathbb{Q}^n , defined by a matrix $\frac{1}{w}A$ where $A = (a_{i,j})_{1 \leq i,j \leq n}$ is an integer matrix and w a positive integer. The Quasi-Linear Transformation or QLT associated with g is the transformation of the discrete plane \mathbb{Z}^n defined by the composition of g with the greatest integer part function noted $\lfloor \cdot \rfloor$. This QLT is then noted G .

In the following we will denote $\delta = \det(A)$, where $\det(A)$ is the determinant of A , and we'll assume that $\delta > 0$. The linear transformation defined by the matrix $\frac{1}{w}A$ extends to \mathbb{R}^n and for each point Y of \mathbb{R}^n , there exists a unique point X of \mathbb{R}^n such that $Y = g(X)$. This is not the same for a QLT. Indeed, each point of \mathbb{Z}^n can have either none, one or several antecedents: The set of antecedents of $X \in \mathbb{Z}^n$ is then called tile with index X .

Definition 2. We call tile with index $X \in \mathbb{Z}^n$ and denote $P_{G,X}$ the set:

$$P_{G,X} = \{Y \in \mathbb{Z}^n \mid G(Y) = X\}$$

Definition 3. We call p -tile or tile of order $p \in \mathbb{N}$, with index $X \in \mathbb{Z}^n$, and denote P_X^p the set: $P_{G^p,X}^p = \{Y \in \mathbb{Z}^n \mid G^p(Y) = X\}$ where G^p denotes the transformation G iterated p times.

Definition 4. Two tiles are geometrically identical if one is the image of the other by a translation of an integer vector.

In this paper we focus on a particular type of QLTs, called "m-determinantal QLTs" and which are defined by $\frac{1}{w}A$ such that w is a multiple of the determinant of A .

Definition 5. A QLT defined by a matrix $\frac{1}{w}A$ such that $w = m \det(A)$ where m is a positive integer, is called a m -determinantal QLT. A 1-determinantal QLT will be called a determinantal QLT.

In the following G will denote a m -determinantal QLT associated with the matrix $\frac{1}{m\delta}A$ where $\delta = \det(A)$, and g the associated rational linear transformation. To simplify the notations, tiles $P_{G,X}$ and $P_{G,X}^p$ will be noted P_X and P_X^p . Moreover $(0, 0, \dots, 0) \in \mathbb{Z}^n$ will be simply noted O . Let denote \widehat{A}^T the transpose the cofactor matrix of A , we have $A^{-1} = \frac{1}{\delta}\widehat{A}^T$. The following proposition allow to determine recursively the p -tiles of a m -determinantal QLT using translations and P_O , the proof can be found in [10].

Proposition 1. *The p -tiles generated by a m -determinantal QLT are all geometrically identical and can be generated recursively by translations of P_O . More precisely, if \mathcal{T}_v refers to the translation of the vector v and if \widehat{A}^T is the transpose of the cofactor matrix of A we have, for all $p \geq 1$:*

$$P_Y^p = \mathcal{T}_{(m\widehat{A}^T)^p Y} P_O^p \tag{1}$$

$$\text{and } P_O^{p+1} = \bigcup_{X \in P_O} \mathcal{T}_{(m\widehat{A}^T)^p X} P_O^p = \bigcup_{X \in P_O^p} \mathcal{T}_{(m\widehat{A}^T) X} P_O \tag{2}$$

Example 1. Figure 1 shows the tile of order 2 of the QLT defined by $\frac{1}{8} \begin{pmatrix} 2 & 3 \\ -2 & 1 \end{pmatrix}$, this tile is composed of 8 tiles of order 1. For each tile of order one we give the index of the tile which is a point of P_O . In figure 2 we see a fractal associated with this QLT. Figure 3 shows the tile of order 12 of the QLT defined by $\frac{1}{2} \begin{pmatrix} -1 & 1 \\ -1 & -1 \end{pmatrix}$ and figure 4 shows the tile of order 7 of the QLT defined by $\frac{1}{2} \begin{pmatrix} 0 & -1 \\ 1 & -1 \end{pmatrix}$.

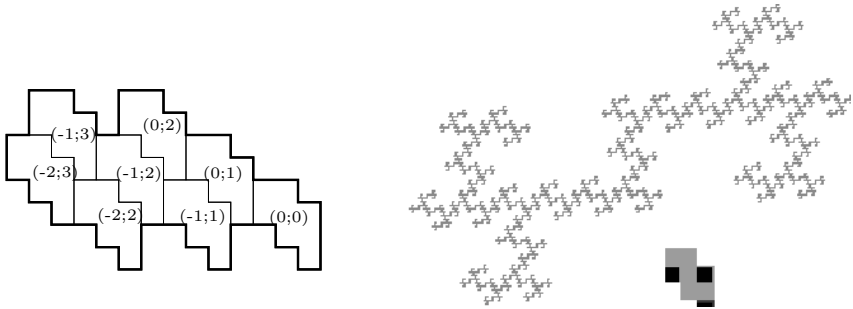


Fig. 1. Tile of order 2 containing 8 tiles of order 1 **Fig. 2.** Fractal associated with QLT order 1

It is well known that if g is a contracting linear transformation of \mathbb{R}^n then g has the origin as unique fixed point and for each $X \in \mathbb{R}^n$ the sequence $g^n(X)$ tends toward this fixed point. We will say that a QLT G is contracting if g is contracting. But such a QLT has not necessarily a unique fixed point. The

behaviour under iteration of 2D contracting QLTs has been studied in [7], [16], [14] and [15]. The following definition and theorem will be used to define new numeration systems.

Definition 6. *A consistent Quasi-Linear Transformation is a QLT which has the origin as unique fixed point such that for each discrete point Y the sequence $(G^n(Y))_{n \geq 0}$ tends toward this unique fixed point.*

Consider a 2D-QLT defined by $\frac{1}{\omega} \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, the infinite norm of g is then

$$\|g\|_\infty = \frac{1}{\omega} \max(|a| + |b|, |c| + |d|).$$

The following theorem, proved in [7], states conditions such that a 2D-QLT is a consistent QLT.

Theorem 1. *Let G be a QLT in \mathbb{Z}^2 such that $\|g\|_\infty < 1$, G is a consistent QLT if and only if one of the three following conditions is verified:*

- (1) $b \leq 0, a + b \leq 0, c > 0$ and $d \leq 0$
- (2) $a \leq 0, b > 0, c \leq 0$ and $c + d \leq 0$
- (3) $a \leq 0, b \leq 0, c \leq 0$ and $d \leq 0$

Remark 1. It would be too long to give here the conditions such that a QLT of norm 1 is consistent. These conditions can be found in [7] and [11] and will be used to define some numeration systems.

The following proposition determines the number of points of a tile of order p and will be used in the last section to determine the fractal dimension of n -dimensional fractals.

Proposition 2. *The number of points of a p -tile generated by a m -determinantal QLT in \mathbb{Z}^n is equal to $\delta^{p(n-1)} m^{np}$.*

Proof. Let first prove that the tile P_O contains exactly $\delta^{n-1} m^n$ points.

Case of $m = 1$ and $n = 2$. It is known (see [7] and [15]) that in this case the number of points of P_O equals δ .

Case of $m = 1$ and $n > 2$. It has been proved in [7] and [3] that for each integer matrix A it exists an unimodular matrix U such that $AU = B$ where B is an upper triangular integer matrix and that the points of the tiles of A are in one-to-one correspondence with the points of the tiles of B . Therefore we will only do the proof for an upper triangular integer matrix $B = (b_{i,j})_{1 \leq i,j \leq n}$, we then have $\delta = b_{1,1} b_{2,2} \dots b_{n,n}$. Let denote P_O , the tile associated with B and P'_X the tiles associated with $B' = (b'_{i,j})_{1 \leq i,j \leq n-1}$ with $b'_{i,j} = b_{i+1,j+1}$ and $\delta' = b_{2,2} b_{3,3} \dots b_{n,n} = \det(B)$. By recurrence hypothesis we will suppose that the number of points of P'_X equals δ'^{n-2} . We have:

$$\begin{aligned}
 & (x_1, x_2, \dots, x_n) \in P_O \\
 \Leftrightarrow & \begin{cases} 0 \leq b_{1,1}x_1 + b_{1,2}x_2 + \dots + b_{1,n}x_n < \delta \\ 0 \leq & b_{2,2}x_2 + \dots + b_{2,n}x_n < \delta \\ \vdots & \vdots \\ 0 \leq & b_{n,n}x_n < \delta \end{cases} \\
 \Leftrightarrow & \begin{cases} -\frac{b_{1,2}x_2 + \dots + b_{1,n}x_n}{b_{1,1}} \leq x_1 < \delta' - \frac{b_{1,2}x_2 + \dots + b_{1,n}x_n}{b_{1,1}} \\ 0 \leq \frac{b_{2,2}x_2 + \dots + b_{2,n}x_n}{\delta'} < b_{1,1} \\ \vdots \\ 0 \leq \frac{b_{n,n}x_n}{\delta'} < b_{1,1} \end{cases} \\
 \Leftrightarrow & \begin{cases} -\frac{b_{1,2}x_2 + \dots + b_{1,n}x_n}{b_{1,1}} \leq x_1 < \delta' - \frac{b_{1,2}x_2 + \dots + b_{1,n}x_n}{b_{1,1}} \\ (x_2, x_3, \dots, x_n) \in P'_{i_2, i_3, \dots, i_n} \text{ with } i_k = 0, 1, \dots, b_{1,1} \text{ for } k = 2, 3, \dots, n \end{cases}
 \end{aligned}$$

The number of solutions for x_1 equals δ' and each tile $P'_{i_2, i_3, \dots, i_n}$ contains δ^{n-2} , therefore we have $\delta' \delta^{n-2} b_{1,1}^{n-1} = \delta^{n-1}$ points.

If $m > 1$, let denote P'_X the tiles associated with $\frac{1}{\delta}A$ and P_O the tile associated with $\frac{1}{m\delta}A$, we have:

$$\begin{aligned}
 & (x_1, x_2, \dots, x_n) \in P_O \\
 \Leftrightarrow & \begin{cases} 0 \leq a_{1,1}x_1 + a_{1,2}x_2 + \dots + a_{1,n}x_n < m\delta \\ \vdots \\ 0 \leq a_{n,1}x_1 + a_{n,2}x_2 + \dots + a_{n,n}x_n < m\delta \end{cases} \\
 \Leftrightarrow & \begin{cases} 0 \leq \frac{a_{1,1}x_1 + a_{1,2}x_2 + \dots + a_{1,n}x_n}{\delta} < m \\ \vdots \\ 0 \leq \frac{a_{n,1}x_1 + a_{n,2}x_2 + \dots + a_{n,n}x_n}{\delta} < m \end{cases} \\
 \Leftrightarrow & (x_1, x_2, \dots, x_n) \in P'_{i_1, i_2, \dots, i_n} \text{ with } i_1, i_2, \dots, i_n = 0, 1, \dots, m-1
 \end{aligned}$$

But each $P'_{i_1, i_2, \dots, i_n}$ contains δ^{n-1} points, it follows that P_O contains $\delta^{n-1}m^n$ points.

We conclude that the number of points of P_O equals $\delta^{n-1}m^n$.

Let now suppose that the number of points of P'_O equals $\delta^{p(n-1)}m^{np}$, we have $P_O^{p+1} = \bigcup_{X \in P_O} \mathcal{T}_{(m\hat{A}^X)^p X} P'_O$ (see proposition [□](#)), it follows that the number of points of P_O^{p+1} equals $\delta^{p(n-1)}m^{np}\delta^{n-1}m^n = \delta^{(p+1)(n-1)}m^{n(p+1)}$. □

3 Quasi-Linear Transformations and Numeration Systems

Let β denote a complex number and \mathcal{D} a finite set of elements of $\mathbb{Z}[\beta]$. (β, \mathcal{D}) is a valid base for $\mathbb{Z}[\beta]$ if each element c of $\mathbb{Z}[\beta]$ can be written uniquely in the form $c = c_0 + c_1\beta + c_2\beta^2 \dots + c_n\beta^n$ with $c_i \in \mathcal{D}$ and $n \in \mathbb{N}$; the length of the decomposition is then $n + 1$. We also say that (β, \mathcal{D}) is a numeration system of $\mathbb{Z}[\beta]$ and \mathcal{D} is the set of digits of this numeration system.

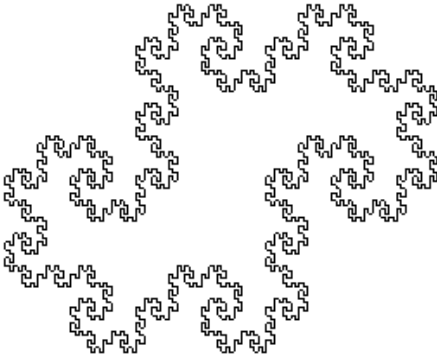


Fig. 3. Border of P_O^{12}

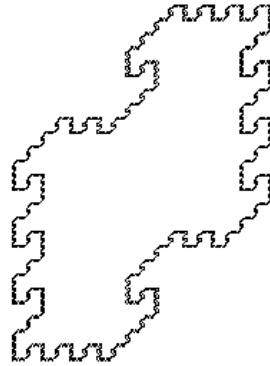


Fig. 4. Border of P_O^7

3.1 Case of Gaussian Integers

The results below can be found in [8]: They allow us to determine numeration systems of $\mathbb{Z}[i]$ by considering $\beta = a + ib$ a Gaussian integer and \mathcal{D} a set of Gaussian integers.

Definition 7. Let $c = x + iy$ and $\beta = a + ib$ be two Gaussian integers. The integer division of c by β , noted $\left\lfloor \frac{c}{\beta} \right\rfloor$, is defined by: $\left\lfloor \frac{c}{\beta} \right\rfloor = \left\lfloor \frac{ax + by}{a^2 + b^2} \right\rfloor + i \left\lfloor \frac{-bx + ay}{a^2 + b^2} \right\rfloor$.

This division corresponds to the usual division of complex numbers composed with the integer part function, so we have the following relation with QLTs.

Proposition 3. Let $\beta = a + ib$ and $c = x + iy$ be two Gaussian integers and let $c' = x' + iy' = \left\lfloor \frac{c}{\beta} \right\rfloor$. The point (x', y') is then the image of the point (x, y) by the QLT G_β defined on \mathbb{Z}^2 by the matrix $\frac{1}{a^2 + b^2} \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$.

The following theorem and its proof can be found in [7] and [8].

Theorem 2. Let $\beta = a + ib$ be a Gaussian integer and \mathcal{D} the set of Gaussian integers c such that $\left\lfloor \frac{c}{\beta} \right\rfloor = 0$, the three following properties are then equivalent:

1. (β, \mathcal{D}) is a numeration system,
2. The QLT G_β is a consistent Quasi-Linear Transformation,
3. $a \leq 0$ and $|a| + |b| > 1$.

Remark 2. Consider the QLT G_β and note P_O and P_O^p the tiles defined by this QLT. In order to determine the set of digits, we have to determine P_O . In [7]

and [9] we can find an algorithm that allows to determine P_O . P_O^p represents the elements of $\mathbb{Z}[\beta]$ such that the decomposition in the numeration system is of length p . For example, figure 3 represents the elements of $\mathbb{Z}[-1 + i]$ such that the decomposition in the numeration system $(-1 + i, \{0, -1\})$ is of length 12.

3.2 Case of Algebraic Integers of Order 2

Now we consider β an algebraic integer such that $\beta^2 + b\beta + a = 0$ with $a, b \in \mathbb{Z}$. We only consider the case where β is a complex number, that is to say $b^2 - 4a < 0$. We will define numeration systems of $\mathbb{Z}[\beta]$ where the digits are elements of $\mathbb{Z}[\beta]$. First we will define an integer division by β , this integer division corresponds to a QLT that will define the digits of the numeration system. We will define the division by using another base of $\mathbb{Z}[\beta]$, the QLT associated with the division will depend on these base, a good choice of the base will raise to a consistent QLT and so define a numeration system.

Remark 3. Let $\beta_1 = q + \beta$ with $q \in \mathbb{Z}$, we have $\mathbb{Z}[\beta] = \mathbb{Z}[\beta_1]$, indeed:

$$\begin{aligned} x &= x_1 + x_2\beta \in \mathbb{Z}[\beta] \\ \Leftrightarrow \exists x_1, x_2 \in \mathbb{Z} \mid x &= x_1 + x_2\beta \\ \Leftrightarrow \exists x_1, x_2 \in \mathbb{Z} \mid x &= x_1 - x_2q + x_2\beta_1 \\ \Leftrightarrow \exists x'_1, x'_2 \in \mathbb{Z} \mid x &= x'_1 + x'_2\beta_1 \\ \Leftrightarrow x \in \mathbb{Z}[\beta_1] \end{aligned}$$

Let now define the integer division using this base. We have $\beta^2 + b\beta + a = 0$, so $\frac{1}{\beta} = -\frac{\beta+b}{a} = \frac{-\beta_1+q-b}{a}$ and $\beta_1^2 = \beta_1(2q-b) - q^2 + qb - a$. Let $x = x_1 + x_2\beta_1$, we have:

$$\begin{aligned} \frac{x}{\beta} &= \frac{(x_1 + x_2\beta_1)(-\beta_1 + q - b)}{a} \\ &= \frac{x_2(\beta_1(b - 2q) + q^2 - qb + a) + \beta_1(x_2(q - b) - x_1) + x_1(q - b)}{a} \\ &= \frac{x_1(q - b) + x_2(q^2 - qb + a) + (-x_1 - x_2q)\beta_1}{a}. \end{aligned}$$

We define the integer division of x by β by the composition of the division above with the integer part function.

Definition 8. Let $x = x_1 + x_2\beta_1$, the quotient of the integer division of x by β , noted $\lfloor \frac{x}{\beta} \rfloor$, is defined by $\lfloor \frac{x}{\beta} \rfloor = \lfloor \frac{x_1(q - b) + x_2(q^2 - qb + a)}{a} \rfloor + \lfloor \frac{-x_1 - x_2q}{a} \rfloor \beta_1$.

Proposition 4. Let $x = x_1 + x_2\beta_1$, and $y = y_1 + y_2\beta_1 = \lfloor \frac{x}{\beta} \rfloor$. The point (y_1, y_2) is then the image of the point (x_1, x_2) by the QLT G_{β_1} defined on \mathbb{Z}^2 by:

$$\frac{1}{a} \begin{pmatrix} q - b & a - qb + q^2 \\ -1 & -q \end{pmatrix}.$$

A digit of the numeration system is an element x of $\mathbb{Z}[\beta]$ such that $\lfloor \frac{x}{\beta} \rfloor = 0$, therefore the set of digits is

$$\mathcal{D} = \{x \in \mathbb{Z}[\beta] \mid \lfloor \frac{x}{\beta} \rfloor = 0\} = \{x = x_1 + x_2\beta_1 \mid G_{\beta_1}(x_1, x_2) \in P_O\}$$

where P_O is the tile of order one associated to the QLT G_{β_1} .

Theorem 3. *Let β be an algebraic integer such that $\beta^2 + b\beta + a = 0$ and \mathcal{D} the set of elements c of $\mathbb{Z}[\beta]$ such that $\lfloor \frac{c}{\beta} \rfloor = 0$, the three following properties are then equivalent:*

1. *There exists q such that (β, \mathcal{D}) is a numeration system,*
2. *There exists q such that the QLT G_{β_1} is a consistent QLT,*
3. *$(b \geq 2)$ or $(b = 1$ and $a \geq 2)$*

Proof. (β, \mathcal{D}) is a numeration system if and only if for all $c \in \mathbb{Z}[\beta]$, there exist $c_0, c_1, \dots, c_p \in \mathcal{D}$ such that $c = c_0 + c_1\beta + \dots + c_p\beta^p$. Moreover, this decomposition has to be unique. It is easy to see that if this decomposition exists then $c_i = x_i + y_i\beta_1$ with $(x_i, y_i) = G_{\beta_1}^i(x, y)$ where $x + y\beta_1 = c$. We conclude that the decomposition exists and is unique if and only if for each $(x, y) \in \mathbb{Z}^2$ there exists $p \in \mathbb{N}$ such that $G_{\beta_1}^p(x, y) = (0, 0)$. Finally (β, \mathcal{D}) is a numeration system if and only if G_{β_1} is a consistent QLT. Theorem 1 gives the necessary and sufficient conditions such that G_{β_1} is a consistent QLT but only for QLTs such that $\|G_{\beta_1}\|_\infty < 1$. The first and the third case of this theorem can not be satisfied. Indeed, in these two cases we have the condition $q^2 - qb + a \leq 0$ and we have assumed that $b^2 - 4a < 0$ so that $q^2 - qb + a$ is always positive. The second case corresponds to the conditions

$$q - b \leq 0, \quad q^2 - qb + a > 0, \quad -1 \leq 0 \quad \text{and} \quad -1 - q \leq 0$$

which is equivalent to $-1 \leq q \leq b$. But $\|G_{\beta_1}\|_\infty < 1$, implies

$$\|G_{\beta_1}\|_\infty = \frac{1}{a} \max(b - q + q^2 - qb + a; 1 + |q|) < 1 \Leftrightarrow \begin{cases} q^2 - (b + 1)q + b < 0 & (1) \\ 1 + |q| < a & (2) \end{cases}$$

If $b \leq 2$, the conditions $-1 \leq q \leq b$ and $q^2 - (b + 1)q + b < 0$ are conflicting. But if $b > 2$, the condition (1) is satisfied for $1 < q < b$ and the condition (2) $1 + q < a$ can always be satisfied (because $b^2 < 4a$).

When $b = 2$ the QLT associated with the integer division is of norm 1. If we choose $q = 2$ the QLT is defined by the matrix $\frac{1}{a} \begin{pmatrix} 0 & a - 1 \\ -1 & -1 \end{pmatrix}$ which is a consistent QLT (see [7], [11]).

When $b = 1$, the QLT associated with the integer division is defined by $\frac{1}{a} \begin{pmatrix} q - 1 & a - q + q^2 \\ -1 & -q \end{pmatrix}$. If $a = 1$ and $q > 1$ the QLT is not contracting, if $a = 1$ and $q = 0$ or 1 the QLT is not consistent (see [7]). If $a > 1$, we can choose $q = 1$, then the QLT is defined by $\frac{1}{a} \begin{pmatrix} 0 & a \\ -1 & -1 \end{pmatrix}$ which is a consistent QLT. □

Example 2. Let $\beta^2 + 3\beta + 3 = 0$, and $\beta_1 = 1 + \beta$, the QLT corresponding to the integer division by β is defined by $\frac{1}{3} \begin{pmatrix} -2 & 1 \\ -1 & -1 \end{pmatrix}$, the set of digits is $\mathcal{D} = \{0, -1, -2 - \beta\}$. If we choose $\beta_1 = 2 + \beta$, the QLT corresponding to the integer division by β is defined by $\frac{1}{3} \begin{pmatrix} -1 & 1 \\ -1 & -2 \end{pmatrix}$, the set of digits is $\mathcal{D} = \{0, -1, -2\}$.

4 Quasi-Linear Transformations and Fractals

4.1 Border of Tiles in 2-Dimension

Let β denote a complex number and \mathcal{D} a finite set of elements of $\mathbb{Z}[\beta]$. (β, \mathcal{D}) is a valid base for $\mathbb{Z}[\beta]$ if each element c of $\mathbb{Z}[\beta]$ can be written uniquely in the form $c = c_0 + c_1\beta + c_2\beta^2 \dots + c_n\beta^n$ with $c_i \in \mathcal{D}$ and $n \in \mathbb{N}$.

Gilbert [6] proved that if (β, \mathcal{D}) is a valid base for $\mathbb{Z}[\beta]$, then every complex number $c \in \mathbb{C}$ has an infinite representation (not necessarily unique) in the form:

$$c = \sum_{j=-\infty}^p c_j \beta^j, \quad c_j \in \mathcal{D}.$$

Let us consider the set of complex numbers with zero integer part in this numeration system (also called "fundamental domain" in the literature [12]), that is to say the set \mathcal{K} of complex numbers c such that:

$$c = \sum_{j=-\infty}^{-1} c_j \beta^j, \quad c_j \in \mathcal{D}.$$

In [5] Gilbert determined the fractal dimension of the border of this set considering the base $\beta = -n + i$ with $n \in \mathbb{N}$ and $\mathcal{D} = 0, 1, \dots, n^2$. Note $K_p = \left\{ c \in \mathbb{C} \mid c = \sum_{j=1}^p \frac{c_j}{\beta^j} \right\}$, the set \mathcal{K} is the limit of K_p when p tends toward infinity.

Denote by ρ and θ the module and argument of β : $\beta = \rho e^{i\theta}$. We have $\rho = \sqrt{\delta}$ where δ is the determinant of the matrix associated to the division, and so:

$$\begin{aligned} K_p &= \left\{ c \in \mathbb{C} \mid c = \sum_{j=1}^p \frac{c_j}{\beta^j} \text{ with } c_j \in \mathcal{D} \right\} \\ &= \left\{ c \in \mathbb{C} \mid c = \frac{1}{\beta^p} \sum_{j=1}^p c_j \beta^{p-j} \text{ with } c_j \in \mathcal{D} \right\} \\ &= \frac{1}{\beta^p} \left\{ c \in \mathbb{C} \mid c = \sum_{j=0}^{p-1} c_j \beta^j \text{ with } c_j \in \mathcal{D} \right\} \end{aligned}$$

If β is a Gaussian integer $\mathcal{D} = P_O$, if β is an algebraic integer

$$\mathcal{D} = \{x = x_1 + x_2\beta_1 \mid (x_1, x_2) \in P_O\}.$$

In [8] we studied particular determinantal QLTs associated with matrices of the form $\frac{1}{a^2+b^2} \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$ and showed how the border of the p-tiles of these QLTs can be generated by a substitution. This study has been generalized to every 2D determinantal QLT, so we define substitutions that generate the border of the p-tiles P_O^p . We don't give the method to generate the substitutions in this paper, it can be found in [11]. The substitution associated with the QLT, determines the border of P_O . If at each step of the substitution we divide the length of these segments by $\sqrt{\delta}$, we obtain the border of a fractal set noted \mathcal{F} (which corresponds exactly to the fundamental domain in case of Gaussian integers). Let denote by N_p the number of segments of P_O^p , the substitution allows to determine N_p . Using the counting box dimension we can determine the fractal dimension of the border which is given by $d = \lim_{p \rightarrow +\infty} \frac{\log N_p}{\log \delta^{\frac{p}{2}}}$.

Example 3. Figure 3 represents a tile associated to the numeration system $(-1 + i, \{0, -1\})$, the associated set K_p defined above tends to the fundamental domain of this numeration system, we obtain the fractal dimension $d = 1,5236$ (which conforms to the result found in [5]). Figure 4 doesn't correspond to a numeration system, the fractal dimension of the set obtained is $d = 1,303$.

4.2 Fractals of \mathbb{Z}^n

Consider the recurrence given in proposition 1 which allows the construction of $P_O^n \in \mathbb{Z}^n$. By applying this recurrence to a subset P'_O of P_O , that is to say that we define

$$P_O^{p+1} = \bigcup_{X \in P'_O} \mathcal{T}_{(m\hat{A}^T)^p X} P_O^p = \bigcup_{X \in P'_O} \mathcal{T}_{(\widehat{m\hat{A}^T}) X} P'_O$$

Proposition 5. *Let denote N_p the number of points of P_O^p and N the number of points removed from P_O to obtain P'_O . We have $N_p = (m^n \delta^{n-1} - N)^p$.*

Proof. In property 2 we proved that the number of points of P_O equals $m^n \delta^{n-1}$ therefore $N_1 = m^n \delta^{n-1} - N$. Suppose that $N_p = (m^n \delta^{n-1} - N)^p$, we have $P_O^{p+1} = \bigcup_{X \in P'_O} \mathcal{T}_{(m\hat{A}^T)^p X} P_O^p$, it follows that:

$$N_{p+1} = N_1 N_p = (m^n \delta^{n-1} - N)(m^n \delta^{n-1} - N)^p = (m^n \delta^{n-1} - N)^{p+1}. \quad \square$$

Like above, at each step of the recurrence, we divide the size of the points by $m\delta^{\frac{n-1}{n}}$. We then obtain a fractal whose box-counting dimension is given by

$$d = \lim_{p \rightarrow \infty} \frac{\log(m^n \delta^{n-1} - N)^p}{\log((m\delta^{\frac{n-1}{n}})^p)} = \frac{\log(m^n \delta^{n-1} - N)}{\log(m\delta^{\frac{n-1}{n}})}$$

If we consider numeration systems, P'_O corresponds to a subset \mathcal{D}' of the set of digits \mathcal{D} , so that the fractal obtained corresponds to the set of numbers with zero integer part and whose decomposition uses only the digits of \mathcal{D}' .

Example 4. In figures 5, 6 and 7 we see P_O and the fractal generated. The black points of P_0 are removed to obtain P'_O (which corresponds to the grey points). The QLT in figure 5 is defined by $\frac{1}{9} \begin{pmatrix} 0 & -3 \\ 3 & 0 \end{pmatrix}$, the fractal dimension of the set is $\frac{\log(7)}{\log(3)} = 1,7712$, it corresponds to the numeration system $(-3i, \{u + iv | u = 0, 1, 2 \ v = 0, -1, -2\})$ and represents the set of numbers with zero integer part and whose decomposition don't use the digits 0 and $2 - 2i$. In figure 6 the QLT is defined by $\frac{1}{13} \begin{pmatrix} -2 & 3 \\ -3 & -2 \end{pmatrix}$, the fractal dimension of the set is $2 \frac{\log(8)}{\log(13)} = 1,6214$, it corresponds to the numeration system $(-2 + 3i, \{0, -1, -2, -3, -4, -1 + i, -2 + i, -3 + i - 4 + i, -2 + 2i, -3 + 2i, -2 - i, -3 - i\})$ and represents the set of numbers with zero integer part and whose decomposition don't use the digits $0, -4, -1 + i, -3 + 2i$ and $-2 - 2i$. In figure 7 the QLT is defined by $\frac{1}{9} \begin{pmatrix} 2 & 3 \\ -2 & 1 \end{pmatrix}$ and the fractal dimension of the set is $\frac{2 \log(5)}{\log(8)} = 1,5479$.

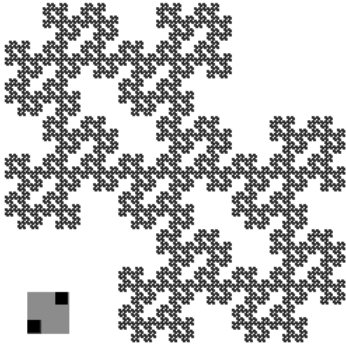


Fig. 5. Fractal associated with numeration systems

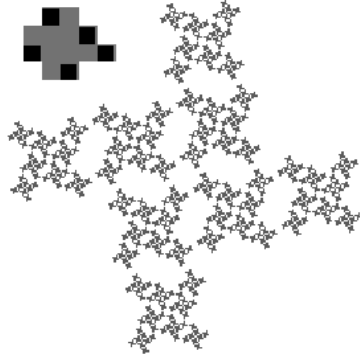


Fig. 6. Another fractal associated with numeration systems

5 Conclusion

We have seen relations that exist between QLTs, numeration systems and fractals. We used consistent 2D-QLTs (see definition 6) to define new numeration systems of $\mathbb{Z}[\beta]$ where β is an algebraic integer of order 2. These numeration systems are associated with 2D-fractals in two ways. If we consider the set of numbers with zero integer part in the numeration system we obtain a set with fractal border. The fractal dimension of this border is determined with help of substitution associated to particular QLTs. This way to obtain fractals is generalised to all m-determinantal 2D QLTs (see definition 5). As a future work, it would be interesting to generalise this method to \mathbb{Z}^n which implies the generalisation of substitutions in higher dimension. If we consider the set of numbers with zero integer part and whose decompositions use a subset of the set of digits

we obtain another fractal set. Properties of the QLTs associated to the numeration systems allow to determine the fractal dimension of these sets. This way to obtain fractal (and their fractal dimension) is generalised to all m -determinantal QLTs in \mathbb{Z}^n . As a future work we would like to generalise the numeration systems to algebraic number of higher order. In [2], the author studied fractals associated with shift radix systems (which generalize numeration systems). Some of these fractals can be obtained using QLTs, one of our future works is to study the relations between QLTs and these numeration systems.

References

- Berthé, V., Siegel, A.: Tilings associated with Beta-Numeration and Substitutions. *Integers: Electronic Journal of Combinatorial Number Theory* 5 (2005)
- Berthé, V., Siegel, A., Steiner, W., Surer, P., Thuswaldner, J.M.: Fractal tiles associated with shift radix systems. *Adv. Math.* (2010) (in press)
- Blot, V., Coeurjolly, D.: Quasi-Affine Transformation in Higher Dimension. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 493–504. Springer, Heidelberg (2009)
- Coeurjolly, D., Blot, V., Jacob-Da Col, M.-A.: Quasi-Affine Transformation in 3-D: Theory and Algorithms. In: Wiederhold, P., Barneva, R.P. (eds.) *IWCIA 2009*. LNCS, vol. 5852, pp. 68–81. Springer, Heidelberg (2009)
- Gilbert, W.J.: The Fractal Dimension of Sets derived from Complex Bases. *Canad. Math. Bull.* (1986)
- Gilbert, W.J.: Complex Based Number Systems. University of Waterloo (2002), <http://www.math.uwaterloo.ca/~wgilbert/Research/FractalPapers.html>
- Jacob, M.-A.: Applications quasi-affines. PhD thesis, Université Louis Pasteur, Strasbourg (1993)
- Jacob, M.-A., Reveilles, J.-P.: Gaussian numeration systems. 5ème colloque DGCI (1995), <https://dpt-info.u-strasbg.fr/~jacob>
- Jacob-Da Col, M.-A.: Applications quasi-affines et pavages du plan discret. *Theoretical Computer Science* 259(1-2), 245–269 (2001), Available in English at <https://dpt-info.u-strasbg.fr/~jacob>
- Jacob-Da Col, M.-A., Tellier, P.: Quasi-linear transformations and discrete tilings. *Theoretical Computer Science* 410, 2126–2134 (2009)
- Jacob-Da Col, M.-A., Tellier, P.: Quasi-linear transformations, substitutions, numerations systems and fractals. Technical Report, Laboratoire LSIT (2010), <https://dpt-info.u-strasbg.fr/~jacob>
- Kátai, I.: Generalized Number Systems in Euclidian Spaces. *Mathematical and Computer Modelling* 38, 883–892 (2003)
- Klette, R., Rosenfeld, A.: Digital straightness—a review. *Discrete Applied Mathematics* 139, 197–230 (2004)
- Nehlig, P., Reveilles, J.-P.: Fractals and Quasi-Affine Transformations. *Computer Graphics Forum* 14, 147–158 (1995)
- Nehlig, P.: Applications quasi-affines: pavages par images réciproques. *Theoretical Computer Science* 156, 1–38 (1995)
- Nehlig, P.: Applications Affines Discrètes et Antialiassage. PhD thesis, Université Louis Pasteur, Strasbourg (1992)
- Reveilles, J.-P.: Géométrie discrète, calcul en nombres entiers et algorithmique. Thèse d’Etat, Université Louis Pasteur, Strasbourg (1991)

Path-Based Distance with Varying Weights and Neighborhood Sequences

Nicolas Normand^{1,2}, Robin Strand³, Pierre Evenou¹, and Aurore Arlicot¹

¹ IRCCyN UMR CNRS 6597, University of Nantes, France

² School of Physics, Monash University, Melbourne, Australia

³ Centre for Image Analysis, Uppsala University, Sweden

Abstract. This paper presents a path-based distance where local displacement costs vary both according to the displacement vector and with the travelled distance. The corresponding distance transform algorithm is similar in its form to classical propagation-based algorithms, but the more variable distance increments are either stored in look-up-tables or computed on-the-fly. These distances and distance transform extend neighborhood-sequence distances, chamfer distances and generalized distances based on Minkowski sums. We introduce algorithms to compute, in \mathbb{Z}^2 , a translated version of a neighborhood sequence distance map with a limited number of neighbors, both for periodic and aperiodic sequences. A method to recover the centered distance map from the translated one is also introduced. Overall, the distance transform can be computed with minimal delay, without the need to wait for the whole input image before beginning to provide the result image.

1 Introduction

In [8] discrete distances were introduced along with sequential algorithms to compute the distance transform (DT) of a binary image, where each point is mapped to its distance to the background. These discrete distances are built from adjacency and connected paths (path-based distances): the distance between two points is equal to the cost of the shortest path that joins them. For distance d_4 (“ d ” in [8]), defined in the square grid \mathbb{Z}^2 , each point has four neighbors located at its top, left, bottom and right edges. Similarly, for distance d_8 (“ d^* ” in [8]), each point has four extra diagonally located neighbors. In both cases, d_4 and d_8 , the cost of a path is defined as the number of displacements. These simple distances have been extended in different ways, by changing the neighborhood depending on the travelled distance ([9,2]), by weighting displacements [5,2], or even by a mixed approach of weighted neighborhood sequence paths [10].

Section 2 presents definitions of distances, disks and some properties of non-decreasing integer sequences that will be used later. Section 3 introduces a new generalization of path-based distances where displacement costs vary both on the displacement vector and on the travelled distance. An application is presented in section 4 for the efficient computation of neighborhood-sequence DT in 2D.

2 Preliminaries

Lambek-Moser inverse of an integer sequence [4]. Let the function f define a non-decreasing sequences of integers $(f(1), f(2), \dots)$. For the sake of simplicity, we call f a sequence. The inverse sequence of f , denoted by f^\dagger , is a non-decreasing sequence of integers defined by:

$$f(m) < n \Leftrightarrow f^\dagger(n) \not\leq m. \tag{1}$$

An interesting property of a sequence f and its inverse f^\dagger is that, by adding the rank of each term to these two sequences, we obtain two complementary sequences $f(m) + m$ and $f^\dagger(n) + n$ [4]. This property extends the results given by Ostrowski *et al.* [7] about Beatty sequences [1]. From [4], we deduce that the inverse of the sequence $f(m) = \lfloor \tau m \rfloor$ with a scalar τ , is $f^\dagger(n) = \lceil \frac{n}{\tau} - 1 \rceil$ so $f(m) + m = \lfloor (1 + \tau)m \rfloor$ and $f^\dagger(n) + n = \lceil (1 + \frac{1}{\tau})n - 1 \rceil$ are two complementary sequences. If τ is irrational, these sequences are Beatty sequences and, for any positive n , $\lceil (1 + \frac{1}{\tau})n - 1 \rceil$ is equal to $\lfloor (1 + \frac{1}{\tau})n \rfloor$ as given in [1].

Proposition 1. $f^\dagger(f(m) + 1) + 1$ is the rank of the smallest term greater than m where f increases.

Proof.

$$\begin{aligned} f^\dagger(f(m) + 1) + 1 = m' &\Leftrightarrow \begin{cases} f^\dagger(f(m) + 1) < m' \\ f^\dagger(f(m) + 1) \geq m' - 1 \end{cases} \\ &\Leftrightarrow \begin{cases} f(m') \geq f(m) + 1 \\ f(m' - 1) < f(m) + 1 \end{cases} \\ &\Leftrightarrow f(m') > f(m) \text{ and } f(m' - 1) \leq f(m). \end{aligned}$$

If we extend f with $f(0) = 0$, and define g by $g(0) = 0, g(n + 1) = f^\dagger(f(g(n)) + 1) + 1$, then $f(g(n))$ takes, in increasing order, all the values of f , each one appearing once.

Definition 1 (Discrete distance). A function $d : \mathbb{Z}^n \times \mathbb{Z}^n \rightarrow \mathbb{N}$ is a translation-invariant distance if the following conditions holds $\forall x, y, z \in \mathbb{Z}^n, \forall \lambda \in \mathbb{Z}$:

1. **translation invariance** $d(x + z, y + z) = d(x, y)$,
2. **positive definiteness** $d(x, y) \geq 0$ and $d(x, y) = 0 \Leftrightarrow x = y$,
3. **symmetry** $d(x, y) = d(y, x)$,

In the following sections, we will drop definiteness and symmetry to define “asymmetric pseudo-distances”.

Definition 2 (Disk). The disk $D(p, r)$ of center p and radius r and the symmetrical disk $\check{D}(p, r)$ are the sets:

$$\begin{aligned} D(p, r) &= \{q : d(p, q) \leq r\}, \\ \check{D}(p, r) &= \{q : d(q, p) \leq r\}. \end{aligned} \tag{2}$$

Table 1. Example of a non-decreasing sequence f and its Lambek-Moser inverse. f is the cumulative sequence of the periodic sequence $(1, 2, 0, 3)$, f^\dagger its inverse. $f^\dagger(f(n) + 1) + 1$ locates the rank of the next f increase. For instance, $f(6) = 9$, $f^\dagger(f(6) + 1) + 1 = 8$ is the rank of appearance of the first value greater than 9, which is 12 in this case.

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$f(n)$	1	3	3	6	7	9	9	12	13	15	15	18	19	21
$f^\dagger(n)$	0	1	1	3	3	3	4	5	5	7	7	7	8	9
$f^\dagger(f(n) + 1) + 1$	2	4	4	5	6	8	8	9	10	12	12	13	14	16

By definition, any disk of negative radius is empty and the disk of radius 0 only contains its center ($D(p, 0) = \{p\}$).

Definition 3 (Distance transform). *The distance transform DT_X of the binary image X is a function that maps each point p to its distance from the closest background point:*

$$DT_X : \mathbb{Z}^n \rightarrow \mathbb{N} \tag{3}$$

$$DT_X(p) = \min \{d(q, p) : q \in \mathbb{Z}^n \setminus X\}.$$

Alternatively, since all points at a distance less than $DT_X(p)$ to p belong to X ($\check{D}(p, DT_X(p) - 1) \subset X$) and at least one point at a distance to p equal to $DT_X(p)$ is not in X ($\check{D}(p, DT_X(p)) \not\subset X$) then:

$$DT_X(p) \geq r \Leftrightarrow \check{D}(p, r - 1) \subset X. \tag{4}$$

The DT is usually defined as the distance *to* the background which is equivalent to the distance *from* the background by symmetry. The equivalence is lost with asymmetric distances, and this definition better reflects the fact that DT algorithms always propagate paths from the background points.

In this paper, we consider path-based distances, *i.e.* distance functions that associate to each couple of points (p, q) , the minimal cost of a path from p to q . For a simple distance, a path is a sequence of points where the difference between two successive points is a displacement vector taken in a fixed neighborhood \mathcal{N} , and the cost (or length) of a path is the number of its displacements. The cost of the path $(p_0, \dots, p_n, p_n + \mathbf{v})$ derives from the cost of the path (p_0, \dots, p_n) :

$$\mathcal{L}(p_0, \dots, p_n) = r \Rightarrow \forall \mathbf{v} \in \mathcal{N}, \mathcal{L}(p_0, \dots, p_n, p_n + \mathbf{v}) = r + 1. \tag{5}$$

Rosenfeld and Pfaltz specifically forbid paths where a point appears more than once [8]. This restriction has no effect on the distance because a path where a point appears more than once can not be minimal. In a similar manner, they exclude the null vector from the neighborhood, forbidding a point to appear several times consecutively. As before, it has no effect on the distance. Notice that, in terms of distance, forbidding a path is equivalent to giving it an infinite cost, so that it can not be minimal. (5) can be rewritten as:

$$\mathcal{L}(p_0, \dots, p_n) = r \Rightarrow \forall \mathbf{v}, \mathcal{L}(p_0, \dots, p_n, p_n + \mathbf{v}) = r + c_{\mathbf{v}},$$

where

$$c_{\mathbf{v}} = \begin{cases} 1 & \text{if } \mathbf{v} \in \mathcal{N} \\ \infty & \text{otherwise} \end{cases} .$$

For a NS-distance characterized by the sequence B :

$$\mathcal{L}(p_0, \dots, p_n) = r \Rightarrow \forall \mathbf{v}, \mathcal{L}(p_0, \dots, p_n, p_n + \mathbf{v}) = r + c_{\mathbf{v}}^B(r), \quad (6)$$

where the displacement cost $c_{\mathbf{v}}^B(r)$ is 1 for a displacement vector in the neighborhood $B(r + 1)$ and infinite otherwise:

$$c_{\mathbf{v}}^B(r) = \begin{cases} 1 & \text{if } \mathbf{v} \in \mathcal{N}_{B(r+1)} \\ \infty & \text{otherwise} \end{cases} \quad (7)$$

For a weighted distance with mask $\mathcal{M} = \{(\mathbf{v}_k; w_k) \in \mathbb{Z}^n \times \mathbb{N}^*\}_{1 \leq k \leq m}$, the distance increment only depends on the displacement vector, but not on the distance already travelled:

$$\mathcal{L}(p_0, \dots, p_n) = r \Rightarrow \forall \mathbf{v}, \mathcal{L}(p_0, \dots, p_n, p_n + \mathbf{v}) = r + c_{\mathbf{v}}, \quad (8)$$

$$c_{\mathbf{v}} = \begin{cases} w & \text{if } (\mathbf{v}; w) \in \mathcal{M} \\ \infty & \text{otherwise} \end{cases}$$

Briefly, the displacement cost for a vector \mathbf{v} and the travelled distance r , is 1 or ∞ , independently of r for simple distances, is equal to 1 or ∞ whether \mathbf{v} belongs or not to $\mathcal{N}_{B(r)}$ for a NS-distance, is in $\mathbb{N}^* \cup \{\infty\}$ according to the chamfer mask and independently of r for a weighted distance.

In the following, we propose to use a displacement cost, denoted by $c_{\mathbf{v}}(r)$, with values in $\mathbb{N}^* \cup \{\infty\}$, that depends both on the displacement vector \mathbf{v} and on the travelled distance r . According to the previous remarks, the cost associated to the null displacement will always be unitary:

$$\forall r \in \mathbb{N}, c_{\mathbf{0}}(r) = 1. \quad (9)$$

3 Path-Based Distance with Varying Weights

Definition 4 (Path). We call path from p to q , any finite sequence of points $P = (p = p_0, p_1, \dots, p_n = q)$ with at least one point, and denote by $\mathcal{P}(p, q)$, the set of these paths.

Notice that this definition of a path is not related to any adjacency relation. The sequence $P = (p)$ is allowed as a path from p to itself. It is distinct from $P = (p, p)$, the path from p to itself with a null displacement.

Definition 5 (Partial and total costs of a path). Let \mathcal{N} be a set of vectors containing the null vector $\mathbf{0}$ and the positive displacement costs $c_{\mathbf{v}}$ (with $c_{\mathbf{0}}(r) = 1$ and $c_{\mathbf{v} \notin \mathcal{N}}(r) = \infty$). The total cost of the path $P = (p_0, p_1, \dots, p_n)$ is:

$$\mathcal{L}(P) = \mathcal{L}_n(P), \quad (10)$$

where $\mathcal{L}_i(P)$ is the partial cost of the path truncated to its $i + 1$ first points (i.e., to its i first displacements):

$$\mathcal{L}_0(P) = \mathcal{L}(p_0) = 0, \tag{11}$$

$$\mathcal{L}_{i+1}(P) = \mathcal{L}(p_0, \dots, p_{i+1}) = \mathcal{L}_i(P) + c_{\mathbf{p}_i \mathbf{p}_{i+1}}(\mathcal{L}_i(P)). \tag{12}$$

Definition 6. We use the notation $C_{\mathbf{v}_k}(r) = r + c_{\mathbf{v}_k}(r)$. $c_{\mathbf{v}_k}(r)$ is the relative cost of the displacement \mathbf{v}_k when the distance travelled so forth is r . $C_{\mathbf{v}_k}(r)$ represents the partial cost of the path after this displacement (the absolute cost of this displacement):

$$\mathcal{L}_{i+1}(P) = \mathcal{L}_i(P) + c_{\mathbf{p}_i \mathbf{p}_{i+1}}(\mathcal{L}_i(P)) = C_{\mathbf{p}_i \mathbf{p}_{i+1}}(\mathcal{L}_i(P)). \tag{13}$$

Definition 7. The pseudo-distance induced by $(\{\mathbf{v}_k\}, c_{\mathbf{v}_k})$ is defined by:

$$d(p, q) = 0 \Leftrightarrow p = q$$

$$d(p, q) = \min_{P \in \mathcal{P}(p, q)} \{ \mathcal{L}(P) \}.$$

Definition 8. We call minimal relative (resp. absolute) cost of displacement, denoted by \hat{c} (resp. \hat{C}), the quantity $\hat{c}_{\mathbf{v}}(r) = \min \{ c_{\mathbf{v}}(s) + s - r, \forall s \geq r \}$ (resp. $\hat{C}_{\mathbf{v}}(r) = \min \{ C_{\mathbf{v}}(s), \forall s \geq r \}$).

Proposition 2 (Preservation of cost order by concatenation). Appending the same displacement to existing paths preserves the relation order of their costs. Let $P = (p_1, \dots, p_{n_P})$ and $Q = (q_1, \dots, q_{n_Q})$ be two paths with costs $\mathcal{L}(P)$ and $\mathcal{L}(Q)$, \mathbf{v} a vector and $P' = (p_1, \dots, p_{n_P}, p_{n_P} + \mathbf{v})$, $Q' = (q_1, \dots, q_{n_Q}, q_{n_Q} + \mathbf{v})$ the extended paths with costs $\mathcal{L}(P')$ and $\mathcal{L}(Q')$ measured with minimal displacement costs. Then:

$$\mathcal{L}(P) \leq \mathcal{L}(Q) \Rightarrow \mathcal{L}(P') \leq \mathcal{L}(Q'). \tag{14}$$

Proof. From (13), $\mathcal{L}(P') = \hat{C}_{\mathbf{v}}(\mathcal{L}(P))$ and $\mathcal{L}(Q') = \hat{C}_{\mathbf{v}}(\mathcal{L}(Q))$. By definition of $\hat{C}_{\mathbf{v}}$, $s \leq r \Rightarrow \hat{C}_{\mathbf{v}}(s) \leq \hat{C}_{\mathbf{v}}(r)$, which gives (14).

Proposition 3. Let $\mathcal{N} = \{ \mathbf{v}_k \}$ be a set of vectors and, $c_{\mathbf{v}}(r)$, the displacement costs for these vectors. There exists a path P from p to q of cost $\mathcal{L}(P) = r$ measured with costs $c_{\mathbf{v}}(r)$ if and only if there exists a path P' from p to q of cost $\mathcal{L}'(P') = r$ measured with the minimal displacement costs $\hat{c}_{\mathbf{v}}(r)$.

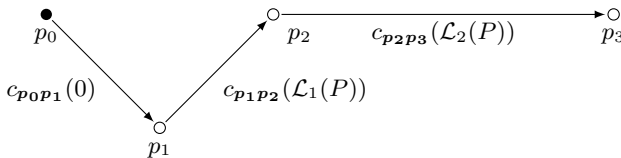


Fig. 1. Total cost of a path $P = (p_0, p_1, p_2)$. Costs of displacements $\mathbf{p}_0 \mathbf{p}_1$, $\mathbf{p}_1 \mathbf{p}_2$ and $\mathbf{p}_2 \mathbf{p}_3$ depend on the partial costs $\mathcal{L}_0(P) = 0$, $\mathcal{L}_1(P) = c_{\mathbf{p}_0 \mathbf{p}_1}(0) + 0$ and $\mathcal{L}_2(P) = c_{\mathbf{p}_1 \mathbf{p}_2}(\mathcal{L}_1(P)) + \mathcal{L}_1(P)$. The total cost of P is $c_{\mathbf{p}_2 \mathbf{p}_3}(\mathcal{L}_2(P)) + \mathcal{L}_2(P)$.

Proof. Consider the cost of P after i displacements, $\mathcal{L}_i(P) = \mathcal{L}_i(p_0, p_1, \dots, p_i)$, we note $m_0 = 1, m_{0 < i \leq n} = 1 + \mathcal{L}_i(P) - \mathcal{L}_{i-1}(P) - \hat{c}_{\mathbf{p}_{i-1}\mathbf{p}_i}(\mathcal{L}_{i-1}(P)) = 1 + c_{\mathbf{p}_{i-1}\mathbf{p}_i}(\mathcal{L}_{i-1}(P)) - \hat{c}_{\mathbf{p}_{i-1}\mathbf{p}_i}(\mathcal{L}_{i-1}(P))$ and $M_i = \sum_{j=0}^i m_j$ the cumulated sum of m_i . Clearly, if $\mathcal{L}(P)$ is finite then each m_i is finite and positive because $\hat{c}_{\mathbf{v}}(r)$ is less than or equal to $c_{\mathbf{v}}(r)$ by construction. Let P' be the (finite) path obtained by m_i occurrences of each point p_i :

$$P' = (p_0, \underbrace{p_1 \dots p_1}_{m_1}, \dots, \underbrace{p_i \dots p_i}_{m_i}, \dots, \underbrace{p_n \dots p_n}_{m_n}).$$

We take as an induction hypothesis that the partial cost of P' after m_i occurrences of p_i , $\mathcal{L}'_{M_{i-1}}(P')$, is equal to $\mathcal{L}_i(P)$. It holds for $i = 0$ because $\mathcal{L}'_{M_0-1}(P') = \mathcal{L}'_{m_0-1}(P') = \mathcal{L}'_0(P') = 0 = \mathcal{L}_0(P)$. If the hypothesis holds for $i - 1$, then the partial cost of P' after the first occurrence of p_i is $\mathcal{L}'_{M_{i-1}}(P') = \mathcal{L}_{i-1}(P) + \hat{c}_{\mathbf{p}_{i-1}\mathbf{p}_i}(\mathcal{L}_{i-1}(P))$, and after $m_i - 1$ repeats of p_i , equals: $\mathcal{L}'_{M_{i-1}+m_i-1}(P') = \mathcal{L}'_{M_{i-1}}(P') = \mathcal{L}_{i-1}(P) + \hat{c}_{\mathbf{p}_{i-1}\mathbf{p}_i}(\mathcal{L}_{i-1}(P)) + m_i - 1 = \mathcal{L}_{i-1}(P) + c_{\mathbf{p}_{i-1}\mathbf{p}_i}(\mathcal{L}_{i-1}(P)) = \mathcal{L}_i(P)$ and the hypothesis is true at rank i . Therefore, for every path of finite cost r measured with \mathcal{L} , there exists a path with the same cost measured with \mathcal{L}' . This is shown in Fig. 2a.

Conversely, let P' be a path with finite cost measured by \mathcal{L}' . We build a path P where each point of P' appears m'_i times consecutively with m'_i such that $m'_i - 1 + c_{\mathbf{p}_i\mathbf{p}_{i+1}}(\mathcal{L}'_i(P')) + m'_i - 1 = \hat{c}_{\mathbf{p}_i\mathbf{p}_{i+1}}(\mathcal{L}'_i(P'))$. By definition of \hat{c} , $\forall r, \exists s : \hat{c}_{\mathbf{v}}(r) = c_{\mathbf{v}}(s) + s - r$, so m'_i exists. Let $M'_0 = 0$ and $M'_{0 < i \leq n} = \sum_{j=0}^{i-1} m'_j$, be the cumulated sum of the previous terms of m'_i .

The induction hypothesis is that the partial cost of P , measured with \mathcal{L} , at the first occurrence of p_i , $\mathcal{L}_{M'_i}(P)$, is equal to $\mathcal{L}'_i(P')$. It holds for $i = 0$ with a null partial cost $\mathcal{L}_{M'_0}(P) = \mathcal{L}_0(P) = 0 = \mathcal{L}'_0(P')$. If the hypothesis holds at rank i , the partial cost of P , after $m'_i - 1$ repetitions of p_i , is $\mathcal{L}_{M'_i+m'_i-1}(P) = \mathcal{L}_{M'_i}(P) + m'_i - 1 = \mathcal{L}'_i(P') + m'_i - 1$, and at the first occurrence of p_{i+1} , equals $\mathcal{L}'_i(P') + m'_i - 1 + c_{\mathbf{p}_i\mathbf{p}_{i+1}}(\mathcal{L}'_i(P') + m'_i - 1) = \mathcal{L}'_i(P') + \hat{c}_{\mathbf{p}_i\mathbf{p}_{i+1}}(\mathcal{L}'_i(P')) = \mathcal{L}'_{i+1}(P')$ and the hypothesis also holds at rank $i + 1$. An example of such a path is shown on Fig. 2b.

Corollary 1. Displacement costs $c_{\mathbf{v}}$ and $\hat{c}_{\mathbf{v}}$ induce the same pseudo-distance.

According to (9), any path from p to q of cost less than r can be extended with null displacements to reach cost r :

$$\mathcal{L}(p_0, \dots, p_n = q) = s < r \Rightarrow \mathcal{L}(p_0, \dots, \underbrace{p_n = q, \dots, q}_{1+r-s}) = r \quad (15)$$

Proposition 4. There exists a path of cost r from p to q if and only if $d(p, q) \leq r$.

Proof. If a path of cost r from p to q exists then by definition of the distance, $d(p, q) = r$ if P cost is minimal, $d(p, q) < r$ otherwise. Conversely, if $d(p, q) = s$ then there exists a path of cost s from p to q that, according to (15), can be extended to cost $r \geq s$.

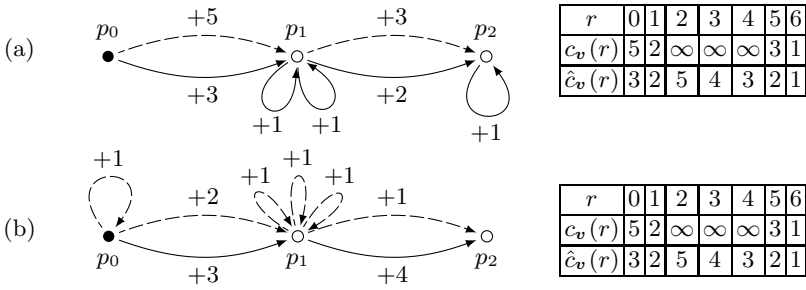


Fig. 2. (a) Given $P = (p_0, p_1, p_2)$, shown with dashed lines, has a total cost $\mathcal{L}(P) = 8$ measured with displacement costs c_v . $P' = (p_0, p_1, p_1, p_1, p_2, p_2)$, solid lines, is built in such a way that its cost $\mathcal{L}'(P')$ measured with minimal displacement costs \hat{c}_v , is equal to $\mathcal{L}(P) = 8$. (b) Given $P' = (p_0, p_1, p_2)$, shown with solid lines, has a total cost $\mathcal{L}'(P') = 7$ measured with displacement costs \hat{c}_v . $P = (p_0, p_0, p_1, p_1, p_1, p_1, p_2)$, dashed lines, is built in such a way that $\mathcal{L}(P) = \mathcal{L}'(P') = 7$.

Corollary 2. For any value of r greater than or equal to $d(p, q)$, there exists a path from p to q which cost is exactly r . The closed disk centered in p with radius r is the set of points for which a path from p of cost equal to r exists:

$$q \in D(p, r) \Leftrightarrow \exists P \in \mathcal{P}(p, q), \mathcal{L}(P) = r. \tag{16}$$

An iterative construction rule of disks is deduced from (16):

$$\begin{aligned} \forall r > 0, D(p, r) &= \bigcup_{v \in \mathcal{N}} \{q : \exists P \in \mathcal{P}(p, q - v) \text{ and } C_v(\mathcal{L}(P)) = r\} \\ &= \bigcup_{\substack{v \in \mathcal{N} \\ s : C_v(s) = r}} D(p + v, s) \end{aligned} \tag{17}$$

4 Minimal Delay Distance Transform

In [12], Wang and Bertrand, proposed a single scan asymmetric generalized DT based on a neighborhood for which there exists a scanning order such that when a point p in the image is scanned, all neighbors of p have already been scanned (forward scan condition). Then, they extended this result to a sequence where two neighborhoods with forward scan condition are alternated (*i.e.*, $B = (1, 2)$) [13]. In the following we propose a method to compute an asymmetric generalized DT based on any number of neighborhoods having forward scan condition used in an arbitrary order defined by a sequence B , either periodic or not. For our purpose, we will use translated versions of regular NS-distances neighborhoods, in order to meet the forward scan condition for each of them. The resulting translated distance map can easily be transformed back into a regular, symmetrical, NS-distance map.

Proposition 5. *The DT of an image X with the distance induced by the neighborhood \mathcal{N} and the displacement costs $C_{\mathbf{v}}$ is such that:*

$$DT_X(p) = \begin{cases} 0 & \text{if } p \notin X \\ \min \{ \hat{C}_{\mathbf{v}}(DT_X(p - \mathbf{v})), \mathbf{v} \in \mathcal{N}^* \} & \text{otherwise} \end{cases} \quad (18)$$

where $\hat{C}_{\mathbf{v}}$ represents the minimal absolute displacement costs corresponding to $C_{\mathbf{v}}$ (definition 8).

Proof. Case $p \notin X$ directly results from definitions 3 and 7. Suppose now that $p \in X$ so any path from $q \notin X$ to p has at least one displacement. Prop. 3 states that distances induced by $(\{\mathbf{v}_k\}, C_{\mathbf{v}_k})$ and $(\{\mathbf{v}_k\}, \hat{C}_{\mathbf{v}_k})$ are equal so we consider the latter cost increments for which prop. 2 holds. According to prop. 2, if $P = (q = p_0, \dots, p_n = p - \mathbf{v})$ is a minimal path from q to $p - \mathbf{v}$ then $P' = (q = p_0, \dots, p_n, p + \mathbf{v})$ has a minimal cost — among paths from q to p with second last point $p - \mathbf{v}$ — equal to $\hat{C}_{\mathbf{v}}(\mathcal{L}(P))$. So $\hat{C}_{\mathbf{v}}(DT_X(p - \mathbf{v}))$ is the shortest distance from a point $q \notin X$ to p via $p - \mathbf{v}$. Since all paths which last displacement \mathbf{v} does not belong to \mathcal{N} have an infinite cost and can not be minimal, (18) holds.

4.1 Generalized Distance Transform

When all vectors in \mathcal{N}^* are directed forward relatively to the scan order, (18) propagates paths from background pixels in a single scan. As a consequence, a generalized DT using any number of neighborhoods $\mathcal{N}_1 \dots \mathcal{N}_n$, selected by a sequence $B, B(i) \in [1, n]$, derives directly from (7, 18) and minimal costs given by:

$$\hat{C}_{\mathbf{v}}(r) = \min \{ s : s > r \text{ and } \mathbf{v} \in \mathcal{N}_{B(s)} \}. \quad (19)$$

let $\chi_{\mathbf{v}}(r)$ denote the characteristic function of the set $\mathcal{N}_{B(r)}$ (i.e., $\chi_{\mathbf{v}}(r) = 1$ if $\mathbf{v} \in \mathcal{N}_{B(r)}$; 0 otherwise) and $\chi_{\mathbf{v}}^{\Sigma}(r)$ its cumulative sum ($\chi_{\mathbf{v}}^{\Sigma}(r) = \sum_{s \leq r} \chi_{\mathbf{v}}(s)$).

Then according to prop. 11

$$\hat{C}_{\mathbf{v}}(r) = [\chi_{\mathbf{v}}^{\Sigma}]^{\dagger}(\chi_{\mathbf{v}}^{\Sigma}(r) + 1) + 1. \quad (20)$$

Algorithm 1 produces a generalized DT using any sequence of neighborhoods (\mathcal{N} represents their union) in forward scan condition, using displacement costs given by (20). A similar algorithm was already presented for the decomposition of convex structuring polygons 6.

4.2 Translated NS-Distance Transform

The sequence of disks for a NS-distance induced by a sequence B is produced by iterative Minkowski sums of neighborhoods:

$$D(p, 0) = \{p\}, \quad D(p, r) = D(p, r - 1) \oplus \mathcal{N}_{B(r)}.$$

For each neighborhood \mathcal{N}_j , we apply a translation vector \mathbf{t}_j such that the translated neighborhood $\mathcal{N}'_j = \mathcal{N}_j \oplus \{\mathbf{t}_j\}$ is in forward scan condition. In a translation

Data: X : a set of points
Data: \mathcal{N} : neighborhood in forward scan condition
Data: \hat{C}_v : minimal absolute displacement costs
Result: DT_X : generalized distance transform of X
foreach p in DT domain, in raster scan **do**
 if $p \notin X$ **then**
 | $DT_X(p) \leftarrow 0$
 else
 | $l \leftarrow \infty$
 foreach v in \mathcal{N} **do**
 | $l \leftarrow \min \{l; \hat{C}_v(DT_X(p - v))\}$
 end
 | $DT_X(p) \leftarrow l$
 end
end

Algorithm 1. Single scan asymmetric distance transform

preserved scan order, t_j translates the first visited point in \mathcal{N}_j to the origin. Assuming a nD standard raster scan order:

$$t_j = (\underbrace{0, \dots, 0}_{n-j}, \underbrace{1, \dots, 1}_j) \tag{21}$$

The translated neighborhoods \mathcal{N}'_1 and \mathcal{N}'_2 obtained with $t_1 = (0, 1)$ and $t_2 = (1, 1)$ are depicted in Fig. 3a and Fig. 3b. Characteristic functions for vectors in $\mathcal{N}'_1 \setminus \mathcal{N}'_2$, $\mathcal{N}'_2 \setminus \mathcal{N}'_1$ and $\mathcal{N}'_1 \cap \mathcal{N}'_2$ (see Fig. 3c-e) are respectively $\mathbf{1}_B$, $\mathbf{2}_B$ and the constant value 1 resulting in the following minimal displacement costs:

$$\hat{C}_v(r) = \begin{cases} \hat{C}_v^1(r) = \mathbf{1}_B^\dagger(\mathbf{1}_B(r) + 1) + 1 & \text{if } v \in \mathcal{N}'_1 \text{ and } v \notin \mathcal{N}'_2 \\ \hat{C}_v^2(r) = \mathbf{2}_B^\dagger(\mathbf{2}_B(r) + 1) + 1 & \text{if } v \notin \mathcal{N}'_1 \text{ and } v \in \mathcal{N}'_2 \\ \hat{C}_v^{12}(r) = r + 1 & \text{if } v \in \mathcal{N}'_1 \text{ and } v \in \mathcal{N}'_2 \end{cases}$$

Periodic sequence. When B is a periodic sequence, minimal relative costs \hat{c}_v are also periodic sequences. Take the periodic sequence of the octagonal distance $B = (\overline{1, 2})$, then $\mathbf{1}_B(r)_{r \geq 0} = (0, 1, 1, 2, \dots)$, $\mathbf{1}_B^\dagger(r)_{r > 0} = (0, 2, 4, \dots)$, $\hat{C}_v^1(r)_{r \geq 0} = (1, 3, 3, 5, \dots)$ and $\hat{c}_v^1(r)_{r \geq 0} = (1, 2, 1, 2, \dots)$. Similarly, $\mathbf{2}_B(r)_{r \geq 0} = (0, 0, 1, 1, 2, \dots)$, $\mathbf{2}_B^\dagger(r)_{r > 0} = (1, 3, \dots)$, $\hat{C}_v^2(r)_{r \geq 0} = (2, 2, 4, \dots)$ and $\hat{c}_v^2(r)_{r \geq 0} = (2, 1, 2, 1, \dots)$.

Rate-based sequence. Suppose now that the sequence of neighborhoods is defined as a Beatty sequence (as in [3]): $B(r) = \lfloor \tau r \rfloor - \lfloor \tau(r - 1) \rfloor$, with $\tau \in [1, 2]$ so that $B(r) \in \{1, 2\}$. $\mathbf{1}_B$ and $\mathbf{2}_B$ are respectively the cumulative sums of $2 - B(r) = \lceil (2 - \tau)r \rceil - \lceil (2 - \tau)(r - 1) \rceil$ and $B(r) - 1 = \lfloor (\tau - 1)r \rfloor - \lfloor (\tau - 1)(r - 1) \rfloor$. Then $\mathbf{1}_B(r) = \lceil (2 - \tau)r \rceil$, $\mathbf{2}_B(r) = \lfloor (\tau - 1)r \rfloor$, $\mathbf{1}_B^\dagger(r) = \lfloor \frac{r-1}{\tau-1} \rfloor$ and $\mathbf{2}_B^\dagger(r) = \lceil \frac{r}{\tau-1} \rceil - 1$. This allows to compute \hat{C}_v^1 and \hat{C}_v^2 on the fly. For the octagonal distance, $\tau = \frac{3}{2}$, $\mathbf{1}_B(r) = \lceil \frac{r}{2} \rceil$, $\mathbf{2}_B(r) = \lfloor \frac{r}{2} \rfloor$, $\mathbf{1}_B^\dagger(r) = 2r - 2$ and $\mathbf{2}_B^\dagger(r) = 2r - 1$.

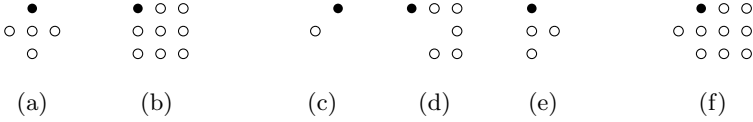


Fig. 3. Neighborhoods used for the translated NS-distance transform. (a), and (b) are respectively the type 1 and 2 translated neighborhoods, \mathcal{N}'_1 and \mathcal{N}'_2 . (c) and (d) and (e) are respectively $\mathcal{N}'_1 \setminus \mathcal{N}'_2$, $\mathcal{N}'_2 \setminus \mathcal{N}'_1$ and $\mathcal{N}'_1 \cap \mathcal{N}'_2$, each set associated to a different sequence of displacement costs. (f) is the whole set of neighbors, $\mathcal{N}'_1 \cup \mathcal{N}'_2$, used for the translated NS-DT.

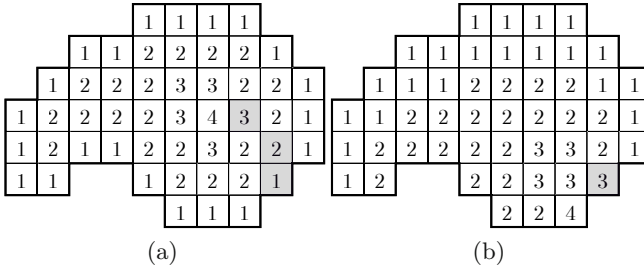


Fig. 4. (a) Octagonal DT of a binary image. (b) Translated octagonal DT. Highlighted centers of disks (a) are translated to the same location, highlighted (b) with value 3.

An example result of algorithm [11](#) for the translated octagonal distance (with displacement costs obtained either from sequence $B = (1, 2)$ either from $\tau = \frac{3}{2}$) is shown in Fig. [4b](#).

4.3 Symmetric DT from Asymmetric DT

Let $\{\mathbf{t}(r), r \in \mathbb{N}^*\}$ be a sequence of translation vectors such that the translated disks $D'(p, r) = D(p + \mathbf{t}(r), r)$ and $\check{D}'(p, r) = \check{D}(p - \mathbf{t}(r), r)$ are increasing according to the set inclusion. For a sequence of disks produced by translated neighborhoods defined in [\(21\)](#), the translation vectors are:

$$\begin{aligned} \mathbf{t}(r) &= \mathbf{t}(r - 1) + \mathbf{t}_{B(r)} \\ &= \sum_j \mathbf{j}_B(r) \mathbf{t}_j \\ &= \left(\sum_{j=n}^n \mathbf{j}_B(r), \dots, \sum_{j=1}^n \mathbf{j}_B(r) \right) \end{aligned}$$

In particular, for the 2D case:

$$\mathbf{t}(r) = (\mathbf{2}_B(r), \mathbf{1}_B(r) + \mathbf{2}_B(r)) = (\mathbf{2}_B(r), r). \tag{22}$$

Data: DT'_X : translated distance map of X

Result: DT_X : centered distance map of X

foreach p in DT' domain **do**

```

|   if  $DT'(p) = 0$  then
|   |    $DT(p) \leftarrow 0$ 
|   else
|   |   foreach  $j$  do
|   |   |    $r \leftarrow \max \{1; DT'(p + \mathbf{t}_j)\}$ 
|   |   |    $r \leftarrow \mathbf{j}_B^\dagger(\mathbf{j}_B(r)) + 1$ ; // First  $r \geq DT'(p - \mathbf{t}_j)$  such that  $B(r) = j$ 
|   |   |   while  $r \leq DT'(p)$  do
|   |   |   |    $DT(p - \mathbf{t}(r - 1)) \leftarrow r$ 
|   |   |   |    $r \leftarrow \mathbf{j}_B^\dagger(\mathbf{j}_B(r)) + 1$ ; // Next  $r$  such that  $B(r) = j$ 
|   |   |   end
|   |   end
|   end
end

```

Algorithm 2. Obtention of a regular (centered) DT from a translated DT

DT'_X has equivalence with values of DT_X :

$$\begin{aligned}
 DT_X(p) \geq r &\Leftrightarrow \check{D}(p, r - 1) \subseteq X \\
 &\Leftrightarrow \check{D}'(p + \mathbf{t}(r - 1), r - 1) \subseteq X \\
 &\Leftrightarrow DT'_X(p + \mathbf{t}(r - 1)) \geq r.
 \end{aligned} \tag{23}$$

Consequently:

$$\begin{aligned}
 DT_X(p) = r &\Leftrightarrow DT_X(p) \geq r \text{ and } DT_X(p) < r + 1 \\
 &\Leftrightarrow DT'_X(p + \mathbf{t}(r)) \leq r \leq DT'_X(p + \mathbf{t}(r - 1)).
 \end{aligned} \tag{24}$$

Knowing $DT'_X(p)$ and $DT'_X(p + \mathbf{t})$, we can deduce the values of $DT_X(p - \mathbf{t}(r - 1))$ for all values of r between $DT'_X(p + \mathbf{t})$ and $DT'_X(p)$ for which $\mathbf{t}(r) = \mathbf{t}(r - 1) + \mathbf{t}$, *i.e.*, $\mathbf{t} = \mathbf{t}_{B(r)}$. Algorithm 2 recovers the values of the centered DT by selecting all r in the interval $[DT'_X(p + \mathbf{t}_j), DT'_X(p)]$ such that $B(r) = j$. Iterating through values r with $B(r) = j$ is achieved using prop. 11. Values of DT'_X become available before the whole image is computed. For instance, in a standard raster scan, as soon as line y is processed, all lines of DT'_X above $y - r_{max}$ are fully recovered (where r_{max} denotes the maximal value of DT' in that line).

5 Conclusion

In this paper, a path-based pseudo-distance scheme where displacement costs vary both with the displacement vector and with the travelled distance was presented. This scheme is generic enough to describe neighborhood-sequence distances, weighted distances as well as generalized distances produced by Minkowski sums. It was shown that a set of displacement costs can be provided in a minimal form, where each displacement vector is assigned a non-decreasing

sequence of costs, without altering the distance function. These non-decreasing sequences are directly applied in the distance transform algorithm to keep track of the costs of minimal paths from the background. An application to a translated neighborhood-sequence distance transform in a single scan was presented along with a method to recover the proper, centered, distance transform. Combined methods provide partial result with a minimal delay, before the input image is fully processed. Their efficiency can benefit all applications where neighborhood-sequence distances are involved, particularly when pipelined processing architectures are involved, or when the size of objects in the source image is limited.

The pseudo-distance presented here is strongly linked to the properties of non-decreasing integer sequences studied by Lambek and Moser. An implementation in C language is publicly available at <http://www.irccyn.ec-nantes.fr/~normand/LUTBasedNSDistanceTransform>.

References

1. Beatty, S.: Problem 3173. *The American Mathematical Monthly* 33(3), 159 (1926)
2. Borgefors, G.: Distance transformations in arbitrary dimensions. *Computer Vision, Graphics and Image Processing* 27(3), 321–345 (1984)
3. Hajdu, A., Hajdu, L.: Approximating the Euclidean distance using non-periodic neighbourhood sequences. *Discrete Mathematics* 283(1-3), 101–111 (2004)
4. Lambek, J., Moser, L.: Inverse and complementary sequences of natural numbers. *The American Mathematical Monthly* 61(7), 454–458 (1954)
5. Montanari, U.: A method for obtaining skeletons using a quasi-Euclidean distance. *Journal of the ACM* 15(4), 600–624 (1968)
6. Normand, N.: Convex structuring element decomposition for single scan binary mathematical morphology. In: Nyström, I., Sanniti di Baja, G., Svensson, S. (eds.) *DGCI 2003*. LNCS, vol. 2886, pp. 154–163. Springer, Heidelberg (2003)
7. Ostrowski, A., Hyslop, J., Aitken, A.C.: Solutions to problem 3173. *The American Mathematical Monthly* 34(3), 159–160 (1927)
8. Rosenfeld, A., Pfaltz, J.L.: Sequential operations in digital picture processing. *Journal of the ACM* 13(4), 471–494 (1966)
9. Rosenfeld, A., Pfaltz, J.L.: Distances functions on digital pictures. *Pattern Recognition* 1(1), 33–61 (1968)
10. Strand, R.: Weighted distances based on neighbourhood sequences. *Pattern Recognition Letters* 28(15), 2029–2036 (2007)
11. Strand, R., Nagy, B., Fouard, C., Borgefors, G.: Generating distance maps with neighbourhood sequences. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) *DGCI 2008*. LNCS, vol. 4992, pp. 295–307. Springer, Heidelberg (2008)
12. Wang, X., Bertrand, G.: An algorithm for a generalized distance transformation based on Minkowski operations. In: *International Conference on Pattern Recognition*, vol. 2, pp. 1164–1168 (1988)
13. Wang, X., Bertrand, G.: Some sequential algorithms for a generalized distance transformation based on Minkowski operations. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14(11), 1114–1121 (1992)

Sparse Object Representations by Digital Distance Functions

Robin Strand

Centre for Image Analysis, Uppsala University, Sweden
robin@cb.uu.se

Abstract. In this paper, some methods for representing objects using path-based distances are considered. The representations can be used as anchor points when extracting medial representations of the objects. The distance transform (DT) is obtained by labeling each object element with the distance to the background. By local operations on the DT, different sets of anchor points can be obtained. We present two different methods based on local operations and prove that the representations are reversible, when this is the case. The methods are defined for weighted distances based on neighborhood sequences, which includes for example the well known cityblock and chessboard distances.

1 Introduction

The medial axis introduced in [3] is an important and often used concept in image processing. The basic idea is to represent the object with its centerline. When applied to digital images, the set of centers of maximal balls (CMBs) is often used when generating a medial axis to guarantee that the original object can be reconstructed. A ball in an object X is *maximal* if it is not contained in any other ball in X . The original object can be recovered from the set of centers of maximal balls (CMBs) together with the corresponding radii. We say that such a representation is *reversible*.

In this paper, we focus on distance functions defined as the minimal cost-path between points. With such path-based distance functions, the distance between two points is calculated by counting the number of (weighted) steps needed to go from one point to the other. We distinguish these *digital* distance functions from the Euclidean metric which is not discrete in this sense. The simplest digital distance functions are the city-block and chessboard distance functions. We consider distance functions defined using both weights (weighted distances) and neighborhood sequences (distances based on neighborhood sequences or ns-distances for short). For the simple distance functions the CMBs appear as local maxima in the distance transform (DT) and are thus easy and fast to extract. For CMBs in the general case with weighted distances or ns-distances, a look-up table is needed in general. In this paper, we present and analyze some natural generalizations of local maxima for weighted ns-distances that are fast and efficient to compute.

Many authors have considered the local-maxima approach, see, e.g., [13,18,18]. For weighted distances, the local maxima in the DT are the points that do not propagate distance information to neighboring grid points when computing the DT. When non-unit weights or a neighborhood sequence of length > 1 is used, there are local maxima that are not CMBs. One approach to overcome this problem is to relabel the distance values, see [11,14]. Relabeling is not enough for all distance functions and another approach is to use look-up tables to extract the set of CMBs, see, e.g., [14,4,12,11]. Another issue is that the set of local maxima and the points that do not propagate distance information when computing the DT are not always equal.

In this paper, some methods for object representation using local operations for weighted ns-distances will be examined. We consider the following representations:

- *Maximal path-points*: The points that do not propagate distance information when computing the DT.
- *Local B-maxima*: When a neighborhood sequence is used to define the distance, the size of the neighborhood at a point \mathbf{p} is given by the neighborhood that is used when computing the DT, can be used to check for local maxima, this is the *local B-maxima*.
- *Local B*-maxima*: Similar to the local B-maxima, but the size of the neighborhood is given by another element in the neighborhood sequence. Local B*-maxima was introduced for ns-distances in [8].
- *Reducing the number of points in a reversible representation*: By removing superfluous points in a reversible representation, a sparse representation is obtained that is still reversible.

In a previous paper, [15], we found that maximal path-points are not well-suited for representing objects for some distance functions. In this paper, we prove that the set MP of maximal path-points can be computed efficiently by a local approach. We also give an alternative method that can be used also for ns-distances. Some of the results on maximal path-points can also be found in [14].

CMBs and local maxima in DTs using the weighted distance have been used for, e.g., skeletonization algorithms [18,2], object decomposition [17], and resolution pyramids [6]. Local maxima in DTs obtained using ns-distances [8] have also been considered in applications such as discrete shading [9] and normal approximation [10].

2 Distance Functions

In this section, some definitions and notions found in, e.g., [14,15,16] are given. Two grid points $\mathbf{p}_1 = (x_1, y_1)$, $\mathbf{p}_2 = (x_2, y_2) \in \mathbb{Z}^2$ are r -neighbors, $r \in \{1, 2\}$, if

$$\begin{aligned} |x_1 - x_2| + |y_1 - y_2| \leq r \quad \text{and} \\ \max\{|x_1 - x_2|, |y_1 - y_2|\} = 1. \end{aligned} \tag{1}$$

The points $\mathbf{p}_1, \mathbf{p}_2$ are *adjacent* if \mathbf{p}_1 and \mathbf{p}_2 are r -neighbors for some r . Two r -neighbors such that the equality in (II) is attained are called *strict r -neighbors*.

Using the notion of 1- and 2-neighbors above, the city-block and chessboard distance functions can be defined. We will consider two ways of generalizing these distance functions by using weights and/or neighborhood sequences.

A *ns B* is a sequence $B = (b(i))_{i=1}^\infty$, where each $b(i)$ denotes a neighborhood relation in \mathbb{Z}^2 . If B is periodic, i.e., if for some fixed strictly positive $l \in \mathbb{Z}_+$, $b(i) = b(i + l)$ is valid for all $i \in \mathbb{Z}_+$, then we write $B = (b(1), b(2), \dots, b(l))$.

The following notation is used for the number of 1:s and 2:s in the ns up to position k .

$$\mathbf{1}_B^k = |\{i : b(i) = 1, 1 \leq i \leq k\}| \quad \text{and} \quad \mathbf{2}_B^k = |\{i : b(i) = 2, 1 \leq i \leq k\}|.$$

A *path* $\mathcal{P}_{\mathbf{p}, \mathbf{q}} = \langle \mathbf{p} = \mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n = \mathbf{q} \rangle$ of length n with start point \mathbf{p}_0 and end point \mathbf{p}_n , is a sequence $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n$ of adjacent grid points. A path is a *B -path* of length n if, for all $i \in \{1, 2, \dots, n\}$, \mathbf{p}_{i-1} and \mathbf{p}_i are $b(i)$ -neighbors.

Definition 1. Given the ns B , the *ns-distance* $d(\mathbf{p}_0, \mathbf{p}_n; B)$ between the points \mathbf{p}_0 and \mathbf{p}_n is the length of (one of) the shortest *B -path(s)* between the points.

Let the real numbers α and β (the *weights*) and a *B -path* $\mathcal{P}_{\mathbf{p}, \mathbf{q}}$ of length n , where exactly l ($l \leq n$) pairs of adjacent grid points in the path are strict 2-neighbors be given. The *cost* of the (α, β) -weighted *B -path* $\mathcal{P}_{\mathbf{p}, \mathbf{q}}$ is $(n - l)\alpha + l\beta$. The *B -path* $\mathcal{P}_{\mathbf{p}, \mathbf{q}}$ between the points \mathbf{p}_0 and \mathbf{p}_n is a *minimal cost (α, β) -weighted B -path between the points \mathbf{p}_0 and \mathbf{p}_n* if no other (α, β) -weighted *B -path* between the points has lower cost than the (α, β) -weighted *B -path* $\mathcal{P}_{\mathbf{p}, \mathbf{q}}$.

Definition 2. Given the ns B and the weights α, β , the *weighted ns-distance* $d_{\alpha, \beta}(\mathbf{p}_0, \mathbf{p}_n; B)$ is the cost of (one of) the *minimal cost (α, β) -weighted B -path(s)* between the points.

Only weights in the interval $\alpha \leq \beta \leq 2\alpha$ are considered.

We write $\mathcal{L}(\mathcal{P}_{\mathbf{p}, \mathbf{q}})$ to denote the length of the path $\mathcal{P}_{\mathbf{p}, \mathbf{q}}$ and $\mathcal{C}_{\alpha, \beta}(\mathcal{P}_{\mathbf{p}, \mathbf{q}})$ to denote the cost of the path $\mathcal{P}_{\mathbf{p}, \mathbf{q}}$. The *concatenation* of two paths $\mathcal{P}_{\mathbf{p}_0, \mathbf{p}_n}$ and $\mathcal{Q}_{\mathbf{q}_0, \mathbf{q}_m}$ such that \mathbf{p}_n and \mathbf{q}_0 are adjacent is $\mathcal{P}_{\mathbf{p}_0, \mathbf{p}_n} \cdot \mathcal{Q}_{\mathbf{q}_0, \mathbf{q}_m} = \langle \mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n, \mathbf{q}_0, \mathbf{q}_1, \dots, \mathbf{q}_m \rangle$.

3 Weighted ns-Distance and Distance Transforms

We state now a functional form of the distance between two grid points $(0, 0)$ and (x, y) in \mathbb{Z}^2 , where $x \geq y \geq 0$. Observe that by translation-invariance and symmetry, the distance between any two grid points is given by the formula presented in Theorem II.

The following theorem is proved in [14, 16]:

Theorem 1. Let the point (x, y) , where $x \geq y \geq 0$, be given. The *weighted ns-distance* between $\mathbf{0}$ and (x, y) is given by

$$d_{\alpha, \beta}(\mathbf{0}, (x, y); B) = (2k - x - y) \cdot \alpha + (x + y - k) \cdot \beta, \\ \text{where } k = \min \{l \in \mathbb{N} : l \geq x + \max(0, y - \mathbf{2}_B^l)\}.$$

When computing the distance transform, this is of course done on a finite subset of the grid, the image domain $\mathcal{I} \subset \mathbb{Z}^2$. The object, a subset of \mathcal{I} is denoted X and its complement in \mathcal{I} is the background \overline{X} . We denote the distance transform for path-based distances with DT_C , where the subscript indicates that costs are computed.

Definition 3. *The distance transform DT_C of an object $X \subset \mathcal{I}$ is the mapping*

$$DT_C : \mathcal{I} \rightarrow \mathbb{R}_0^+ \text{ defined by}$$

$$\mathbf{p} \mapsto d(\mathbf{p}, \overline{X}), \text{ where}$$

$$d(\mathbf{p}, \overline{X}) = \min_{\mathbf{q} \in \overline{X}} \{d(\mathbf{p}, \mathbf{q})\}.$$

For weighted ns-distances, the size of the neighborhood allowed in each step is determined by the *length* of the minimal cost-paths (not the cost), so this value is also needed when propagating distance information. We define the transform $DT_{\mathcal{L}}$ that holds the length of a minimal cost-path at each point.

Definition 4. *The transform $DT_{\mathcal{L}}$ of an object X is the mapping*

$$DT_{\mathcal{L}} : \mathcal{I} \rightarrow \mathbb{N} \text{ defined by}$$

$$\mathbf{p} \mapsto d_{1,1}(\mathbf{p}, \mathbf{q}; B), \text{ where}$$

$$\mathbf{q} \text{ is such that } d_{\alpha,\beta}(\mathbf{q}, \mathbf{p}; B) = d_{\alpha,\beta}(\mathbf{p}, \overline{X}; B).$$

In [14], algorithms (proved to give correct results) for computing DT_C and $DT_{\mathcal{L}}$ are given. The transforms DT_C and $DT_{\mathcal{L}}$ are illustrated in Figure 1. In the figures, each grid point is represented by the corresponding Voronoi region (picture element or *pixel*). The algorithms in [14] can also be used to compute the reverse DT, which is used to obtain the original object from the reversible object representation.

4 Object Representation by Maximal Path-Points

When extracting CMBs for weighted distances, there is a correspondence between the points that do not propagate distance information and the local maxima in the DT, [12]. We will now define this correspondence also for the distances that are defined by neighborhood sequences. A distance propagating path is a path along which local distance values can be propagated when computing the DT. See Figure 1.

Definition 5. *Given an object grid point $\mathbf{p} \in X$ and a background grid point $\mathbf{q} \in \overline{X}$, the minimal cost B -path $\mathcal{P}_{\mathbf{q},\mathbf{p}} = \langle \mathbf{q} = \mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n = \mathbf{p} \rangle$ is a distance propagating B -path if*

- (i) $\mathcal{C}_{\alpha,\beta}(\langle \mathbf{p}_0, \dots, \mathbf{p}_i \rangle) = DT_C(\mathbf{p}_i)$ for all i and
- (ii) $\mathbf{p}_i, \mathbf{p}_{i+1}$ are $b(DT_{\mathcal{L}}(\mathbf{p}_i) + 1) - neighbors$ for all i .

In an object X , the end-point of a distance propagating path of maximal length is called a *maximal path-point*, defined below. A local B -maximum is a local maximum obtained by using the neighborhood that is used to propagate distance information at that point, i.e., the neighborhood defined by $b(DT_{\mathcal{L}}(\mathbf{p}) + 1)$ at the point \mathbf{p} .

Definition 6. *A point $\mathbf{p} \in X$ is called a maximal path-point if there is a $\mathbf{q} \in \overline{X}$ and a path $\mathcal{P}_{\mathbf{q},\mathbf{p}} = \langle \mathbf{q} = \mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n = \mathbf{p} \rangle$ such that the following statements both hold:*

- $\mathcal{P}_{\mathbf{q},\mathbf{p}}$ is a distance propagating B -path of length n defining $DT_{\mathcal{C}}(\mathbf{p})$ and
- there is no \mathbf{p}' such that $\langle \mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n = \mathbf{p}, \mathbf{p}' \rangle$, i.e., $\mathcal{P}_{\mathbf{q},\mathbf{p}} \cdot \langle \mathbf{p}' \rangle$ is a minimal cost (α, β) -weighted B -path of length $n + 1$ such that $DT_{\mathcal{C}}(\mathbf{p}') = \mathcal{C}_{\alpha,\beta}(\langle \mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n = \mathbf{p}, \mathbf{p}' \rangle)$.

Also, the path $\mathcal{P}_{\mathbf{q},\mathbf{p}}$ is called a maximal path.

We denote the set of maximal path-points by MP . It is clear that the set of maximal path-points are the points that do not propagate distance information to neighboring grid points when computing $DT_{\mathcal{C}}$.

Definition 7. *Let the ns B and the weights (α, β) such that $\alpha \leq \beta \leq 2\alpha$ be given. A point $\mathbf{p} \in X$ is a local B -maximum if for all its 1-neighbors \mathbf{r}^i :*

$$DT_{\mathcal{C}}(\mathbf{r}^i) < DT_{\mathcal{C}}(\mathbf{p}) + \alpha$$

and, if $b(DT_{\mathcal{L}}(\mathbf{p}) + 1) = 2$, for all its strict 2-neighbors \mathbf{r}^j :

$$DT_{\mathcal{C}}(\mathbf{r}^j) < DT_{\mathcal{C}}(\mathbf{p}) + \beta.$$

In Theorem 3, we will prove that under some conditions the set of local B -maxima and the set MP are equivalent. To check if a grid point \mathbf{p} is a local B -maximum, the size of the neighborhood at \mathbf{p} is used. Since the size of the neighborhood at \mathbf{p} is determined by $DT_{\mathcal{L}}(\mathbf{p})$ and $DT_{\mathcal{L}}$ in its original form is not unique for some objects, we need to put an additional constraint on $DT_{\mathcal{L}}$. This problem is illustrated in the following example.

Example 1. Consider $B = (1, 2)$ and $(\alpha, \beta) = (2, 3)$. In Figure 1(a) and (b), two minimal cost $(2, 3)$ -weighted B -paths from \overline{X} to the point \mathbf{p} with the distance value 10 are shown. Since they are of different lengths, different neighborhoods are considered at \mathbf{p} .

We will prove that, when $DT_{\mathcal{L}}$ satisfies the following definition, then the set of maximal path-points and the set of local B -maxima are equivalent.

Definition 8. *Given an object X , a ns B and weights α, β , we say that $DT_{\mathcal{L}}$ associated with $DT_{\mathcal{C}}$ holds information about the smallest neighborhoods if it holds information about the minimal cost-path with smallest size of the neighborhood at each point of X .*

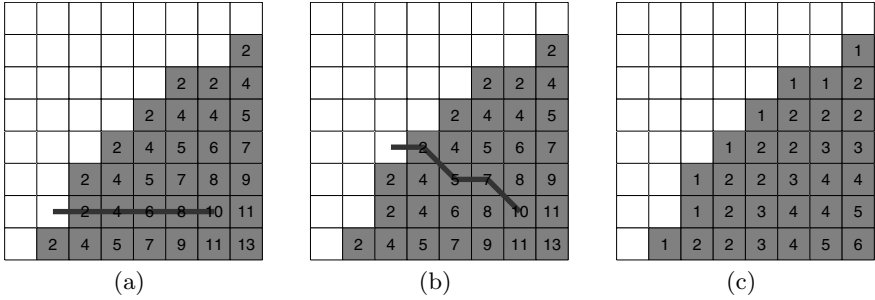


Fig. 1. (a,b): DT_C with two distance propagating paths overlaid, both containing the point labeled 10, defining different neighborhoods at the point. (c): DT_C With information about the smallest neighborhoods. The ns $B = (1, 2)$ and weights $\alpha = 2$ and $\beta = 3$ are used.

Let the ball obtained by weighted ns-distances with radius $r \in \mathbb{R}^+$ be $\mathcal{B}(\mathbf{p}, r) = \{\mathbf{q} : d_{\alpha,\beta}(\mathbf{p}, \mathbf{q}; B) < r\}$. The main theorem from [15] is that the set MP together with the corresponding distance values is a reversible representation of the object:

Theorem 2. *If either*

$$\alpha < \beta \leq 2\alpha \text{ or} \tag{2}$$

$$\alpha = \beta \text{ and } (B = (1) \text{ or } B = (2)), \tag{3}$$

then $\forall \mathbf{q} \in X \setminus MP, \exists \mathbf{p} \in MP : \mathbf{q} \in \mathcal{B}(\mathbf{p}, DT_C(\mathbf{p}))$.

We now give a more efficient way of computing the set MP . We will see that, given a point \mathbf{p} , it is enough to consider a small neighborhood of that point to determine if it is a maximal point or not. This will be done using the notion of local B -maxima:

Theorem 3. *If DT_C holds information about the smallest neighborhoods is used in Definition 7, then Definition 7 and Definition 6 define the same set of points.*

Proof. Let $\mathbf{p} \in X$ be a maximal path-point. Then there is a $\mathbf{q} \in \overline{X}$ and a distance propagating B -path $\mathcal{P}_{\mathbf{q},\mathbf{p}}$ such that any $b(\mathcal{L}(\mathcal{P}_{\mathbf{q},\mathbf{p}}) + 1)$ -neighbor \mathbf{r} to \mathbf{p} is such that

$$DT_C(\mathbf{r}) < \mathcal{C}_{\alpha,\beta}(\mathcal{P}_{\mathbf{q},\mathbf{p}} \cdot \langle \mathbf{r} \rangle) = \mathcal{C}_{\alpha,\beta}(\mathcal{P}_{\mathbf{q},\mathbf{p}}) + \omega,$$

where ω is α if \mathbf{p}, \mathbf{r} are 1-neighbors and β if \mathbf{p}, \mathbf{r} are strict 2-neighbors. This holds for any neighborhood smaller than or equal to $b(\mathcal{L}(\mathcal{P}_{\mathbf{q},\mathbf{p}}) + 1)$, so it holds for the smallest neighborhood. This implies that \mathbf{p} is a local B -maximum.

If, on the other hand, $\mathbf{p} \in X$ is a local B -maximum, then there is a $\mathbf{q} \in \overline{X}$ and a distance propagating B -path $\mathcal{P}_{\mathbf{q},\mathbf{p}}$ with smallest neighborhood at \mathbf{p} such that $DT_C(\mathbf{p}) = \mathcal{C}_{\alpha,\beta}(\mathcal{P}_{\mathbf{q},\mathbf{p}})$. For this path, any $b(\mathcal{L}(\mathcal{P}_{\mathbf{q},\mathbf{p}}) + 1)$ -neighbor \mathbf{r} to \mathbf{p} is such that

$$DT_C(\mathbf{r}) < \mathcal{C}_{\alpha,\beta}(\mathcal{P}_{\mathbf{q},\mathbf{p}} \cdot \langle \mathbf{r} \rangle) = \mathcal{C}_{\alpha,\beta}(\mathcal{P}_{\mathbf{q},\mathbf{p}}) + \omega,$$

where ω is α if \mathbf{p}, \mathbf{r} are 1-neighbors and β if \mathbf{p}, \mathbf{r} are 2-neighbors. Therefore, \mathbf{p} is also a maximal path-point. \square

The Algorithm [1](#) below can be applied to all distance functions presented here. It is a one-scan algorithm. It follows that it is linear in time. Note that for ns-distances (when $(\alpha, \beta) = (1, 1)$), the set of maximal path-points together with the radii is in general *not* a reversible representation of X .

Algorithm 1. Algorithm for finding the set of maximal path-points

Input: $B, (\alpha, \beta), DT_C$, the DT_C that holds information about the smallest neighborhoods

Output: The set MP

For each point \mathbf{p} in X : Assign \mathbf{p} to MP if the conditions in Definition [7](#) are fulfilled.

5 Local B^* -Maxima

In the previous section, we used an approach inspired by how algorithms that compute DTs work – the points that do not propagate distance information are stored as the object representation. In this section, an approach similar to the local B -maxima will be used. A point \mathbf{p} in an object X is a local B^* -maximum if it satisfies the following definition:

Definition 9. Let the ns B and the weights (α, β) such that $\alpha \leq \beta \leq 2\alpha$ be given. A point $\mathbf{p} \in X$ is a local B^* -maximum if for all its 1-neighbors \mathbf{r}^i :

$$DT_C(\mathbf{r}^i) < DT_C(\mathbf{p}) + \alpha$$

and, if $b(DT_C(\mathbf{p})) = 2$, for all its strict 2-neighbors \mathbf{r}^j :

$$DT_C(\mathbf{r}^j) < DT_C(\mathbf{p}) + \beta.$$

Note that the only difference between Definition [7](#) and [9](#) is that another neighborhood is used to decide when strict 2-neighbors are allowed for finding local maxima. Opposed to the local B -maxima, the local B^* -maxima can be used to derive reversible representations for ns-distances.

Lemma 1. Let $\mathcal{P}_{\mathbf{p}, \mathbf{q}}$ be a minimal cost-path between \mathbf{p} and \mathbf{q} .

If $2_B^{\mathcal{L}(\mathcal{P}_{\mathbf{p}, \mathbf{q}})} > 2_{\mathcal{P}_{\mathbf{p}, \mathbf{q}}}$ and \mathbf{q} and \mathbf{r} are 2-neighbors, then

$$d_{\alpha, \beta}(\mathbf{p}, \mathbf{r}; B) \leq d_{\alpha, \beta}(\mathbf{p}, \mathbf{q}; B) + \beta.$$

Proof. Since $2_B^{\mathcal{L}(\mathcal{P}_{\mathbf{p}, \mathbf{q}})} > 2_{\mathcal{P}_{\mathbf{p}, \mathbf{q}}}$, there is at least one element 2 in the neighborhood sequence that does not correspond to a strict 2-step in the path $\mathcal{P}_{\mathbf{p}, \mathbf{q}}$. Therefore, there is a point \mathbf{r}' that is 1-neighbor with both \mathbf{q} and \mathbf{r} and a minimal cost-path $\mathcal{P}_{\mathbf{p}, \mathbf{r}'}$ obtained by swapping a 1-step in $\mathcal{P}_{\mathbf{p}, \mathbf{q}}$ to a 2-step. We have

$$d_{\alpha, \beta}(\mathbf{p}, \mathbf{r}'; B) \leq d_{\alpha, \beta}(\mathbf{p}, \mathbf{q}; B) + \beta - \alpha.$$

Now, by adding a 1-step to $\mathcal{P}_{\mathbf{p}, \mathbf{r}'}$, a B -path between \mathbf{p} and \mathbf{r} of cost $d_{\alpha, \beta}(\mathbf{p}, \mathbf{q}; B) + \beta$ is obtained. \square

Lemma 2. *Given two arbitrary points \mathbf{w} and \mathbf{p} .*

– *If \mathbf{q} and \mathbf{p} are 1-neighbors, then*

$$\mathcal{B}(\mathbf{p}, r) \subset \mathcal{B}(\mathbf{q}, r + \alpha),$$

where $r = d_{\alpha, \beta}(\mathbf{w}, \mathbf{p}; B)$.

– *If \mathbf{q} and \mathbf{p} are strict 2-neighbors and there is a minimal cost B -path $\mathcal{P}_{\mathbf{w}, \mathbf{p}}$ between the points such that $b(\mathcal{L}(\mathcal{P}_{\mathbf{w}, \mathbf{p}})) = 2$. Then*

$$\mathcal{B}(\mathbf{p}, r) \subset \mathcal{B}(\mathbf{q}, r + \beta),$$

where $r = d_{\alpha, \beta}(\mathbf{w}, \mathbf{p}; B)$.

Proof. Consider the case when \mathbf{q} and \mathbf{p} are 1-neighbors. Let $\mathbf{r} \in \mathcal{B}(\mathbf{p}, r)$, i.e., $d_{\alpha, \beta}(\mathbf{r}, \mathbf{p}; B) < d_{\alpha, \beta}(\mathbf{w}, \mathbf{p}; B)$. We will now prove that, when the conditions in the lemma is fulfilled, $\mathbf{r} \in \mathcal{B}(\mathbf{q}, r + \alpha)$.

By adding a 1-step to the path defining $d_{\alpha, \beta}(\mathbf{r}, \mathbf{p}; B)$, a B -path between \mathbf{r} and \mathbf{q} of length $d_{\alpha, \beta}(\mathbf{r}, \mathbf{q}; B) + \alpha$ is obtained. Thus,

$$d_{\alpha, \beta}(\mathbf{r}, \mathbf{q}; B) \leq d_{\alpha, \beta}(\mathbf{r}, \mathbf{p}; B) + \alpha < d_{\alpha, \beta}(\mathbf{w}, \mathbf{p}; B) + \alpha,$$

so $\mathbf{r} \in \mathcal{B}(\mathbf{q}, r + \alpha)$.

Now we focus on the case when \mathbf{q} and \mathbf{p} are 2-neighbors. Let $\mathbf{r} \in \mathcal{B}(\mathbf{p}, r)$, i.e.,

$$d_{\alpha, \beta}(\mathbf{r}, \mathbf{p}; B) < d_{\alpha, \beta}(\mathbf{w}, \mathbf{p}; B). \tag{4}$$

The path $\mathcal{P}_{\mathbf{r}, \mathbf{p}}$ is a minimal cost B -path between \mathbf{r} and \mathbf{p} . The path $\mathcal{P}_{\mathbf{w}, \mathbf{p}}$ is a minimal cost B -path between \mathbf{w} and \mathbf{p} such that $b(\mathcal{L}(\mathcal{P}_{\mathbf{w}, \mathbf{p}})) = 2$ by the conditions in the lemma.

★ **Case i** $\mathcal{L}(\mathcal{P}_{\mathbf{r}, \mathbf{p}}) \geq \mathcal{L}(\mathcal{P}_{\mathbf{w}, \mathbf{p}})$ (not possible when $\alpha = \beta$, since then the path-length is proportional to the path-cost)

Since $\mathcal{P}_{\mathbf{r}, \mathbf{p}}$ is longer than $\mathcal{P}_{\mathbf{w}, \mathbf{p}}$, but has lower cost, it follows that $2_B^{\mathcal{L}(\mathcal{P}_{\mathbf{r}, \mathbf{p}})} > 2_{\mathcal{P}_{\mathbf{r}, \mathbf{p}}}$.
Now

$$\begin{aligned} d_{\alpha, \beta}(\mathbf{r}, \mathbf{q}; B) &\leq d_{\alpha, \beta}(\mathbf{r}, \mathbf{p}; B) + \beta \quad (\text{by Lemma 1}) \\ &< d_{\alpha, \beta}(\mathbf{w}, \mathbf{p}; B) + \beta. \quad (\text{by (4)}) \end{aligned}$$

Therefore, $\mathbf{r} \in \mathcal{B}(\mathbf{q}, r + \beta)$

★ **Case ii** $\mathcal{L}(\mathcal{P}_{\mathbf{r}, \mathbf{p}}) < \mathcal{L}(\mathcal{P}_{\mathbf{w}, \mathbf{p}})$

• **Case ii-a** $\mathcal{L}(\mathcal{P}_{\mathbf{r}, \mathbf{p}}) = \mathcal{L}(\mathcal{P}_{\mathbf{w}, \mathbf{p}}) - 1$

Since $b(\mathcal{L}(\mathcal{P}_{\mathbf{w}, \mathbf{p}})) = 2$

$$d_{\alpha, \beta}(\mathbf{r}, \mathbf{q}; B) \leq d_{\alpha, \beta}(\mathbf{r}, \mathbf{p}; B) + \beta < d_{\alpha, \beta}(\mathbf{w}, \mathbf{p}; B) + \beta.$$

Therefore, $\mathbf{r} \in \mathcal{B}(\mathbf{q}, r + \beta)$ also in this case.

• **Case ii-b** $\mathcal{L}(\mathcal{P}_{\mathbf{r}, \mathbf{p}}) \leq \mathcal{L}(\mathcal{P}_{\mathbf{w}, \mathbf{p}}) - 2$

▶ **Case ii-b-1** $d_{\alpha, \beta}(\mathbf{r}, \mathbf{p}; B) \leq d_{\alpha, \beta}(\mathbf{w}, \mathbf{p}; B) - 2\alpha + \beta$

$$\begin{aligned} d_{\alpha, \beta}(\mathbf{r}, \mathbf{q}; B) &\leq d_{\alpha, \beta}(\mathbf{r}, \mathbf{p}; B) + 2\alpha \quad \text{independent of } B \\ &\leq d_{\alpha, \beta}(\mathbf{w}, \mathbf{p}; B) + \beta \quad \text{since Case ii-b-1} \end{aligned}$$

Therefore, $\mathbf{r} \in \mathcal{B}(\mathbf{q}, r + \beta)$

► **Case ii-b-2** $d_{\alpha,\beta}(\mathbf{r}, \mathbf{p}; B) > d_{\alpha,\beta}(\mathbf{w}, \mathbf{p}; B) - 2\alpha + \beta$

In this case, we have $2_{\mathcal{P}_{\mathbf{r},\mathbf{p}}} > 2_{\mathcal{P}_{\mathbf{w},\mathbf{p}}}$ and since $\mathcal{L}(\mathcal{P}_{\mathbf{r},\mathbf{p}}) < \mathcal{L}(\mathcal{P}_{\mathbf{w},\mathbf{p}})$, we have that $2_B^{\mathcal{L}(\mathcal{P}_{\mathbf{r},\mathbf{p}})} > 2_{\mathcal{P}_{\mathbf{r},\mathbf{p}}}$. Using Lemma 1, we get

$$d_{\alpha,\beta}(\mathbf{r}, \mathbf{q}; B) \leq d_{\alpha,\beta}(\mathbf{r}, \mathbf{p}; B) + \beta < d_{\alpha,\beta}(\mathbf{w}, \mathbf{p}; B) + \beta,$$

so $\mathbf{r} \in \mathcal{B}(\mathbf{q}, r + \beta)$. □

When applied to a $DT_{\mathcal{L}}$ that holds information about the smallest neighborhoods, Lemma 2 proves the following theorem:

Theorem 4. $\forall \mathbf{r}$ in X that is not a local B^* -maximum, there is a local B^* -maximum $\mathbf{p} : \mathbf{r} \in \mathcal{B}(\mathbf{p}, DT_{\mathcal{L}}(\mathbf{p}))$.

It follows that the linear-time Algorithm 2 below can be used to extract reversible representations of an object X .

Algorithm 2. Algorithm for finding the set of local B^* -maxima.

Input: $B, (\alpha, \beta), DT_{\mathcal{L}}$, the $DT_{\mathcal{L}}$ that holds information about the smallest neighborhoods

Output: The set of local B^* -maxima

For each point \mathbf{p} in X : Assign \mathbf{p} to MP if the conditions in Definition 9 are fulfilled.

6 Reducing the Cardinality of Object Representations

Let the set of points in the object representation be MR . In [5], an algorithm that can be used to reduce the set MR is presented. The idea is to iteratively remove the maximal path-points that correspond to balls that are covered by the union of all other balls in MR . This is done by, for each $\mathbf{p} \in X$, computing how many balls $\mathcal{B}(\mathbf{q}, DT_{\mathcal{L}}(\mathbf{q}))$ with center $\mathbf{q} \in MR$ that meet \mathbf{p} . If all points in a ball $\mathcal{B}(\mathbf{q}, DT_{\mathcal{L}}(\mathbf{q}))$ with center $\mathbf{q} \in MR$ meet at least two balls, then \mathbf{q} can be removed from MR . Repeating this procedure for increasing distance values gives a representation of X that consist of few points. We call the obtained representation the *reduced MR*. Note that the reduced MR obtained in this way might not be the optimal set, [7].

7 Examples

In Figure 2, local B -maxima (computed by Algorithm 1), local B^* -maxima (computed by Algorithm 2), and sets of CMBs are shown using some distance functions. CMBs can be extracted by, e.g., exhaustive search or by look-up tables, [14][12][11]. The object in Figure 2 is not perfectly recovered by the local B -maxima for ns-distances, since it is not a reversible representation for ns-distances.

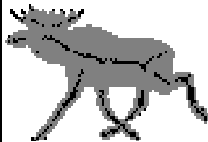
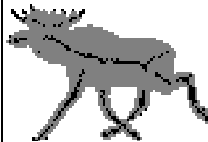













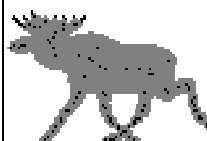








local B -maxima (the set MP)	local B^* -maxima	CMBs	Reduced CMB
$B = (1), (\alpha, \beta) = (1, 1)$			
 222 elements	 222 elements	 222 elements	 177 elements
$B = (2), (\alpha, \beta) = (1, 1)$			
 250 elements	 250 elements	 250 elements	 161 elements
$B = (1, 2), (\alpha, \beta) = (1, 1)$			
 418 elements	 295 elements	 295 elements	 157 elements
$B = (2), (\alpha, \beta) = (3, 4)$			
 453 elements	 453 elements	 317 elements	 118 elements
$B = (1, 2), (\alpha, \beta) = (2, 3)$			
 451 elements	 406 elements	 278 elements	 131 elements
$B = (1, 2, 1, 2, 2), (\alpha, \beta) = (4, 5)$			
 471 elements	 435 elements	 289 elements	 126 elements

Fig. 2. Object representations for some weighted ns-distances. The object contains 1640 elements. The recovered object is shown for each set of object representations.

8 Conclusions and Future Work

We have presented two methods based on local neighborhoods that can be used for distance functions defined by a neighborhood sequence and weights:

- Maximal path-points (introduced in [15]) defined as the points that do not propagate distance information when computing the DT. We proved that the set of maximal path-points MP can be computed efficiently by finding local B -maxima.
- Local B^* -maxima.

Both methods are fast, since the methods both use local neighborhoods.

If the weights α and β are equal, then the method based on maximal path-points does not give reversible representations. Note that, when a constant neighborhood sequence is used, the two approaches are equal, see Figure 2.

The object representation based on local B^* -maxima was used for ns-distances (i.e., without weights) in [8]. The proof in [8] is based on closed balls and the conditions for the non-weighted case are the same as the ones used here.

An interesting observation is that, for ns-distances, the set of local B^* -maxima is equivalent to the set of CMBs in Figure 2. We are not aware of any verification that this is true in general, but it seems likely, so we leave this as a conjecture.

The quality of the representations for weighted ns-distances using the local operations presented here (local B -maxima and local B^* -maxima) shown in Figure 2 can be improved. It is well-known, [11,14], that when non-unit weights are used, the distance values in the DT should be relabeled. In [11], it is proved that when using the weights $\alpha = 3$ and $\beta = 4$ and a constant ns $B = (2)$, then the set of local maxima in a relabeled DT equals the set of CMBs. In [14], object representations obtained by relabeling of general weighted ns-distances combined with local B -maxima (i.e., maximal path-points) is described. The number of elements is significantly reduced by this relabeling.

Experimental results have shown that the object representation obtained by relabeling DT_C with weighted ns-distances combined with local B^* -maxima is very similar to the set of CMBs. We thus have some promising, preliminary result showing that the number of points in the object representation is significantly reduced when relabeling is used. It seems that reducing the object representation in this way does not affect the reversibility. Therefore, we expect our future research to generate strong results on object representations using relabeled DTs.

References

1. Arcelli, C., Sanniti di Baja, G.: Finding local maxima in a pseudo-Euclidean distance transform. *Computer Vision, Graphics, and Image Processing* 43, 361–367 (1988)
2. Sanniti di Baja, G., Thiel, E.: Skeletonization algorithm running on path-based distance maps. *Image and Vision Computing* 14(1), 47–57 (1996)

3. 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)
4. Borgfors, G.: Centres of maximal discs in the 5-7-11 distance transform. In: Proceedings of 8th Scandinavian Conference on Image Analysis (SCIA 1993), Tromsø, Norway, pp. 105–111 (1993)
5. Borgfors, G., Nyström, I.: Efficient shape representation by minimizing the set of centres of maximal discs/spheres. *Pattern Recognition Letters* 18, 465–472 (1997)
6. Borgfors, G., Ramella, G., Sanniti di Baja, G., Svensson, S.: On the multiscale representation of 2D and 3D shapes. *Graphical Models and Image Processing* 61(1), 44–62 (1999)
7. Coeurjolly, D., Hulin, J., Sivignon, I.: Finding a minimum medial axis of a discrete shape is NP-hard. *Theoretical Computer Science* 406(1-2), 72–79 (2008)
8. Kumar, M.A., Chatterji, B.N., Mukherjee, J., Das, P.P.: Representation of 2D and 3D binary images using medial circles and spheres. *International Journal of Pattern Recognition and Artificial Intelligence* 10(4), 365–387 (1996)
9. Mukherjee, J., Kumar, M.A., Chatterji, B.N., Das, P.P.: Discrete shading of three-dimensional objects from medial axis transform. *Pattern Recognition Letters* 20(14), 1533–1544 (1999)
10. Mukherjee, J., Kumar, M.A., Das, P.P., Chatterji, B.N.: Use of medial axis transforms for computing normals at boundary points. *Pattern Recognition Letters* 23(14), 1649–1656 (2002)
11. Normand, N., Évenou, P.: Medial axis LUT computation for chamfer norms using H-polytopes. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) *DGCI 2008. LNCS*, vol. 4992, pp. 189–200. Springer, Heidelberg (2008)
12. Rémy, E., Thiel, E.: Medial axis for chamfer distances: computing look-up tables and neighbourhoods in 2D or 3D. *Pattern Recognition Letters* 23, 649–661 (2002)
13. Rosenfeld, A., Pfaltz, J.L.: Sequential operations in digital picture processing. *Journal of the ACM* 13(4), 471–494 (1966)
14. Strand, R.: Distance Functions and Image Processing on Point-Lattices: with focus on the 3D face- and body-centered cubic grids. Ph.D. thesis, Uppsala University, Sweden (November 2008)
15. Strand, R.: Shape representation with maximal path-points for path-based distances. In: Proceedings of 5th International Symposium on Image and Signal Processing and Analysis (ISPA 2007), Istanbul, Turkey. pp. 397–402 (2007)
16. Strand, R.: Weighted distances based on neighbourhood sequences. *Pattern Recognition Letters* 28(15), 2029–2036 (2007)
17. Svensson, S., Sanniti di Baja, G.: Using distance transforms to decompose 3D discrete objects. *Image and Vision Computing* 20(8), 529–540 (2002)
18. Svensson, S., Borgfors, G., Nyström, I.: On reversible skeletonization using anchor-points from distance transforms. *Journal of Visual Communication and Image Representation* 10(4), 379–397 (1999)

Efficient Robust Digital Hyperplane Fitting with Bounded Error

Dror Aiger, Yukiko Kenmochi, Hugues Talbot, and Lilian Buzer

Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge, France

Abstract. We consider the following fitting problem: given an arbitrary set of N points in a bounded grid in dimension d , find a digital hyperplane that contains the largest possible number of points. We first observe that the problem is 3SUM-hard in the plane, so that it probably cannot be solved exactly with computational complexity better than $O(N^2)$, and it is conjectured that optimal computational complexity in dimension d is in fact $O(N^d)$. We therefore propose two approximation methods featuring linear time complexity. As the latter one is easily implemented, we present experimental results that show the runtime in practice.

Keywords: fitting, digital hyperplane, approximation, linear programming, randomization.

1 Introduction

This article is about efficient and effective hyperplane fitting in the presence of outliers, formulated in the discrete setting. This is a well-known problem with many applications in computer vision and image analysis. In the following we consider an arbitrary set of pixels S in discrete space \mathbb{Z}^d , (with typically $d = 2$ or 3), and ask the question of what is the closest co-dimension 1 hyperplane (*i.e.* a line in 2D or a plane in 3D) that best fits this set of pixels. Clearly this problem depends on the notion of fitting and the definition of hyperplane.

Irrespective of the formulation, recent applications of this problem include shape approximation [3,19], image registration [18,20] and image segmentation [12,14]. More generally, the problem of robust parameter estimation from noisy data is also closely related [10].

The problem is connected to the problem of robust regression, for instance using least squares, weighted least-squares, least-absolute-value regression and least median of squares (LMS) [4,15,17]. In this setting, an hyperplane \mathbf{H} of co-dimension 1 corresponds to the following continuous model:

$$\mathbf{H} = \{(x_1, x_2, \dots, x_d) \in \mathbb{R}^d : a_1x_1 + a_2x_2 + \dots + a_dx_d + a_{d+1} = 0\}, \quad (1)$$

with the $a_i \in \mathbb{R}$. Note it is common to add an additional normalization constraint such as $\sum_{i=1}^d |a_i| = 1$ or $\sum_{i=1}^d a_i^2 = 1$. Such a formulation corresponds to minimizing various cost functions. For instance least-square fitting, which has a closed-form solution, minimizes the geometric distance from all the given points

to the model, whereas least-absolute-value regression optimizes the ℓ_1 distance instead. Both models are very efficient but not very robust to outliers. In contrast, LMS minimizes the median of either the geometric or ℓ_1 distance to the model, and is robust to the presence of outliers, as long as they do not represent more than half of the dataset. There exists polynomial, optimal algorithms for LMS, but their space and computational complexity grow at least as quickly as $O(N^d)$, and there does not seem to exist dependable implementations for $d \geq 3$.

1.1 Approximate Line and Plane Fitting

Well-known and most used methods for hyperplane fitting include the Hough Transform (HT) [6][11], RANSAC [7] and associated variations [5]. HT uses an accumulative approach in the discretized dual space and an ad-hoc detection of high accumulated values. The computational complexity of traditional HT is $O(N\delta^{d-1})$, where N is the number of given points and δ is the (discrete) size of image sides. This is linear for a fixed δ and a fixed d .

RANSAC and its variation consider random d -tuples of points (*i.e.* pairs and triplets respectively for 2D or 3D) within the pixel set S , forming respectively a candidate hyperplane, and compute a distance from S to the candidate hyperplane. Many distances can be considered, including robust versions that may or may not be true distances. A distance or score is associated with every candidate hyperplane. After a number of candidate trials that depend on the size of S , the best-fitting hyperplane featuring the minimum distance or score is given as the output. The computational complexity of RANSAC is linear in the size N of S if the number of inliers is a constant fraction of N .

Both HT and RANSAC are efficient, linear-time complexity algorithms. However neither can claim to find a global or even error-bounded approximate solution to an associated optimization problem.

1.2 Discrete, Optimal Formulation

Because the problem is inherently discrete, it is useful to consider a purely discrete formulation of the problem.

Recently, such a discrete, optimization-based framework for this problem was proposed in [24] for 2D lines and 3D planes. This approach consists of considering a discrete line or plane using Reveillès definition [16], *i.e.* a set of grid points between a pair of continuous-domain lines or planes with rational coefficients separated by a distance w . For a given pixel set S of size N and a given w , an optimal hyperplane is one for which a maximal number of pixels lie between the two continuous hyperplanes. Even though the pair of hyperplanes form a convex set, the problem is combinatorial in nature, and a non-polynomial, branch-and-bound approach was initially suggested to find the optimal solution. This was later solved with an $O(N^d \log N)$ algorithm for $d = 2, 3$ in [21][22][23], and improved in [13] with an $O(N^2)$ solution in 2D using a topological sweep method. While a polynomial solution of degree equal to the dimension of the problem is useful, it is still too inefficient for many application. Typically, the problem is currently solvable for $N = 10^3$ but impractical for $N = 10^6$ in 3D.

In the rest of the paper we first show that the problem is 3SUM-hard in 2D, *i.e.* that a computational complexity of $O(N^2)$ is likely the best that can be expected for 2D, as well as $O(N^d)$ for higher dimension d . This motivated us to pursue an approximate solution with known bounded error, with linear-time complexity. We present two algorithms that give us different bounded errors; one is the number of inliers, and another is the width of digital hyperplanes. Since algorithms based on HT and RANSAC provide no error bound, and no guarantee of optimality of any kind, this render our solution competitive with RANSAC and HT, while insuring a good quality solution. We demonstrate this on a pair of artificial and real examples.

2 The Problem of Digital Hyperplane Fitting

2.1 Definition

An hyperplane in Euclidean space \mathbb{R}^d , $d \geq 2$, is defined by (I), and in this paper we use the following normalization constraint instead of the above more conventional ones: $-1 \leq a_i \leq 1$ for $i = 1, 2, \dots, d$ such that there exists at least one a_i that equals to 1. Note that this normalization technique enables us to bound the range of every coefficient between -1 and 1 , except for a_{d+1} : practically a_{d+1} is also bounded by the size of an input image.

A digital hyperplane, which is the digitization of a hyperplane \mathbf{H} in the discrete space \mathbb{Z}^d , is then defined by the set of discrete points satisfying two inequalities:

$$\mathbf{D}(\mathbf{H}) = \{(x_1, x_2, \dots, x_d) \in \mathbb{Z}^d : 0 \leq a_1x_1 + a_2x_2 + \dots + a_dx_d + a_{d+1} < w\} \quad (2)$$

where w is a given constant. From the digital geometrical viewpoint, if we set $w = 1$, then the definition is equivalent to that of naive hyperplanes [16], where parameters can be rational numbers. As mentioned above, at least one coefficient a_i equals 1 among a_i for $i = 1, 2, \dots, d$, so hereafter we set $a_d = 1$ for simplicity. In other words, we mainly deal with the following linear inequalities

$$0 \leq a_1x_1 + a_2x_2 + \dots + a_{d-1}x_{d-1} + x_d + a_{d+1} < w \quad (3)$$

instead of the ones in (2). Note that practically we use d different types of the inequalities for representing digital hyperplanes depending on d settings of $a_i = 1$ for all $i = 1, \dots, d$.

2.2 Problem Formulation

Given a set S of discrete points coming from the $[0, \delta]^d$ grid (*i.e.* the hypercubic grid with a distance between two neighbours = 1), the problem is to find a digital hyperplane that contains a maximum number of points, called an optimal digital hyperplane. Discrete points that are contained in the fitted digital hyperplane, are called inliers; the complement points are called outliers. Our problem is then equivalent to finding a digital hyperplane such that the number of inliers be maximum.

We need the following definition for geometric understanding:

Definition 1. *A slab is the region on and in between two parallel hyperplanes in d -space. An ω -slab is a slab of size ω , meaning that the distance between the two hyperplanes is ω .*

If the distance ω is taken in the x_d -axis direction in the space (x_1, x_2, \dots, x_d) and $\omega = w$, then $\mathbf{D}(\mathbf{H})$ can be geometrically interpreted as a w -slab. Thus, finding the optimal digital hyperplane for a given S is equivalent to finding a w -slab that contains the maximum number of points in S . Using standard geometric duality induced by (3), the problem of finding this w -slab is equivalent to finding a point that is covered by a maximum number of w -slabs in the dual space as follows: every point \mathbf{p} in the d -dimensional primal space (x_1, x_2, \dots, x_d) is mapped to a hyperplane \mathbf{H} in the d -dimensional dual space $(a_1, a_2, \dots, a_{d-1}, a_{d+1})$ and the set of all hyperplanes in the primal that have a distance (in the x_d -axis direction) smaller than w to \mathbf{p} are mapped to the set of points in the dual contained in a w -slab (distance w in the a_{d+1} -axis direction) one of whose sides is equal to \mathbf{H} . Details for the $d = 2$ case can be found in [13].

In the following sections, the problem of digital hyperplane fitting will be considered either in the primal space (Sections 3 and 5) or in the dual space (Section 4).

3 Theoretical Observation on Exact Fitting

We consider the $[0, \delta]^d$ grid, and a set S of N discrete points is given. As our input is a binary image, N is necessarily smaller than δ^d . We show in this section that, for any N that is smaller than δ^d , the exact solution of the fitting problem is as hard to obtain as that of $O(N^d)$ problems; for $d = 2$, we call such a class of problems the 3SUM problem. For the sake of simplicity, we start our discussion in the 2D plane.

3SUM is the following computational problem, introduced by Gajentaan and Overmars [9] and conjectured to require roughly quadratic time complexity: given a set T of n integers, are there elements a, b, c in T such that $a + b + c = 0$? A problem is called 3SUM-hard if solving it in subquadratic time implies a subquadratic time algorithm for 3SUM.

Observation 1. *The problem of digital line fitting is 3SUM-hard.*

Proof. The problem of finding three colinear points among a given set of discrete points was proven to be 3SUM-Hard in [9]. We now reduce our problem of digital line fitting to the problem of finding three colinear points.

Given a set S of discrete points, we can compute the value δ corresponding to the length of the bounding box of S in linear time. For three points $\mathbf{a} = (x_a, y_a)$, $\mathbf{b} = (x_b, y_b)$, $\mathbf{c} = (x_c, y_c)$ such that $x_a \neq x_b$, the vertical distance between a straight line l_{ab} going through \mathbf{a} and \mathbf{b} and a point \mathbf{c} is given by

$d(l_{ab}, \mathbf{c}) = \frac{|(\mathbf{c}-\mathbf{a}) \wedge (\mathbf{b}-\mathbf{a})|}{|(0,1) \wedge (\mathbf{b}-\mathbf{a})|}$. Note that \wedge is the exterior product between two vectors, which is an algebraic generalization of the cross product between two 3D vectors to any d dimension (here, $d = 2$), so that the result is a bivector. As we process integer coordinates, if we consider three non colinear points, we thus know that $|(\mathbf{c} - \mathbf{a}) \wedge (\mathbf{b} - \mathbf{a})| \geq 1$. Then, we know that $|(0, 1) \wedge (\mathbf{b} - \mathbf{a})| = |x_{\mathbf{a}} - x_{\mathbf{b}}| \leq \delta$ by definition of δ . Therefore, we can conclude that for any three non-colinear points $\mathbf{d}, \mathbf{e}, \mathbf{f}$ of S , we have: $d(l_{de}, \mathbf{f}) \geq \frac{1}{\delta}$. We can scale the points of S in linear time by an integer factor δ in order to obtain the new set S' . Then, any three non-colinear points $\mathbf{d}', \mathbf{e}', \mathbf{f}'$ of S' must satisfy $d(l_{d'e'}, \mathbf{f}') \geq 1$.

To conclude, if our algorithm for finding an optimal digital line detects at least three points $\mathbf{d}, \mathbf{e}, \mathbf{f}$ that can be covered by a digital line, this means that they satisfy $d(l_{de}, \mathbf{f}) < 1$. However, as we know that in S' , three non colinear points would generate a thickness equal to or greater than 1, this means that these three points are colinear. Thus, using our algorithm and a linear-time reduction, we can detect if S contains three colinear points, therefore our problem is 3SUM-hard.

The extension of 3SUM problem to higher dimensions is considered as the problem of detecting affine degeneracy of a given collection of N hyperplanes, *i.e.* finding a subset of $d + 1$ hyperplanes intersecting in a common point. This is conjectured to require $O(N^d)$ time. In this paper, we do not seek a proof since the case for $d = 2$ is sufficient for our purpose, as we seek a linear algorithm.

As solving the problem exactly in arbitrary dimension is likely at least quadratic, we next suggest two approximations. The first has a theoretical proved characteristics but is not easy to implement. The second is a more practical algorithm that features a proven worst case runtime and can be easily implemented. Moreover, its practical runtime can be much better than what is suggested by its worst case analysis.

4 Approximation with Bounded Error in Number of Inlier Points

In this section, we show that if the optimal number of inlier points is not too small (*i.e.* $\Omega(N)$), an approximation of the optimal digital hyperplane can be found in linear time, with respect to N and the runtime also depends on the given approximation value ε . We use the dual space and make a simple use of the tool to solve approximately the problem of linear programming with violations, presented in [2]. In this approximation, we do not use the fact that the points lie on a grid and it is correct for any set of points in \mathbb{R}^d .

We start with the results of Aronov et al. [2] on linear programming with violations. They obtained a randomized algorithm which is correct with high probability. Afshani et al. also [1] obtained a Las Vegas algorithm (*i.e.* one that either provides the correct answer or informs about failure).

Theorem 1 (Aronov et al. [2]). *Let L be a linear program with n constraints in \mathbb{R}^d , and let f be the objective function to be minimized. Let k_{opt} be the minimum number of constraints that must be violated to make L feasible, and let \mathbf{v} be the point minimizing $f(\mathbf{v})$ with k_{opt} constraints violated. Then one can output a point $\mathbf{u} \in \mathbb{R}^d$ such that \mathbf{u} violates at most $(1 + \varepsilon)k_{opt}$ constraints of L , and $f(\mathbf{u}) \leq f(\mathbf{v})$. The results returned are correct with high probability. The expected running time (which also holds with high probability) of this algorithm is $O(n + \varepsilon^{-4} \log n)$ for $d = 2$, and $O(n(\varepsilon^{-2} \log n)^{d+1})$ for larger d .*

In our case, we are only interested in finding a point in the dual space that is covered by the maximum number of w -slabs. We reduce this problem to the problem of linear programming with violations and solve it using the result of [2]. The following observation is a result of replacing each w -slab with two halfspaces that have the w -slab as their intersection, represented by (3). We thus have $2N$ constraints and a point in space is violating (*i.e.*, not covered by) k w -slabs among N , if and only if it is violating k halfspaces among the $2N$. Therefore, the tool for finding a point violating the minimum number of halfspaces finds also a point that is covered by the maximum number of w -slabs. Let n_{opt} be the maximum possible number of w -slabs that can contain a point in d -space (*i.e.*, inliers) and let k_{opt} the optimal number of violations, $N = k_{opt} + n_{opt}$. For the approximation parameter, we observe that since we find a point that violates at most $(1 + \varepsilon)k_{opt}$ slabs, we actually find a point that is covered by at least $N - (1 + \varepsilon)k_{opt}$ w -slabs. Let n be the number of points that are covered by the w -slab we just found. If $n_{opt} = \Omega(N)$ we have $n > (1 - c\varepsilon)n_{opt}$ for some fixed c .

We now observe the following:

Observation 2. *Given a set of N points in the d -dimensional space and some $\varepsilon > 0$, we can find a w -slab that contains at least $(1 - \varepsilon)n_{opt}$ points, where n_{opt} is the maximum possible number of points that can be found in a w -slab, assuming $n_{opt} = \Omega(N)$. The runtime is $O(N + \varepsilon^{-4} \log N)$ for $d = 2$ and $O(N(\varepsilon^{-2} \log N)^{d+1})$ for larger d .*

This result can be immediately used for our original problem of finding the optimal digital hyperplane by using the set of grid points.

5 Approximation with Bounded Error in Digital Hyperplane Width

In this section, we show another kind of approximation that makes use of the fact that the input points are in a bounded grid as well. The meaning of this approximation is slightly different from the previous one. The advantage of this version is that it is easy to implement, unlike the previous one that is mainly of theoretical interest and is probably hard to implement. Moreover, using this approximation, we can do both: prove its worst case runtime and also allow to get optimal solution with practical good runtime and/or to have an approximation to any desired level in better runtime than the worst case. We need the following result from Fonseca and Mount [8].

Theorem 2 (Fonseca and Mount [8]). *For a set of N points in the unit d -dimensional cube and some $\varepsilon > 0$, one can build a data structure with $O(\varepsilon^{-d})$ storage space, in $O(N + \varepsilon^{-d} \log^{O(1)}(\varepsilon^{-1}))$ time, such that for a given query hyperplane \mathbf{H} , the number of points on and below \mathbf{H} can be approximately reported in $O(1)$ time, in the following sense: all the points (below \mathbf{H}) that have a larger distance¹ than ε from \mathbf{H} are counted. Points that are closer to \mathbf{H} on both sides may or may not be reported.*

See the next section for the detail of building a specially-designed data structure for the query.

We can now state and prove the main theorem of this section:

Theorem 3. *Given a set of N points on a grid $[0, \delta]^d$, and some $\varepsilon > 0$, $w > 0$, a digital hyperplane of width $w + 5\varepsilon$ that contains $n > n_{opt}$ points, can be found in $O(N + (\frac{\delta}{\varepsilon})^d \log^{O(1)}(\frac{\delta}{\varepsilon}))$ time where n_{opt} is the maximum number of points that any digital hyperplane of width w in $[0, \delta]^d$ can contain.*

Proof. For simplicity, we can assume that all points are given in the unit d -dimensional cube such that their coordinates are integer multiplication of $1/\delta$. Let $w' = w/\delta$ be the new width and $\varepsilon' = \varepsilon/\delta$ be the new approximation parameter. We first build the data structure for halfspace range count in $O(N + \varepsilon'^{-d} \log^{O(1)}(\varepsilon'^{-1}))$ time (Theorem 2) and then we query all $O(\varepsilon'^{-d})$ hyperplanes twice (every digital hyperplane is a w -slab that is the intersection of two halfspaces) in constant time for each one of them.

For the approximation, we will consider the 2-dimensional case, which is simpler to describe. Our algorithm queries all $(w' + 3\varepsilon')$ -slabs by querying two parallel lower halfplanes of distance $w' + 3\varepsilon'$ and subtract their returned numbers. As seen in Figure 11, any optimal digital line must be contained between two such parallel lines of distance $w' + 3\varepsilon'$, \mathbf{Q}_1 , \mathbf{Q}_2 . Let n_{opt} be the optimal number of points contained in the optimal digital line. Let n_1 be the number of points returned by querying a line \mathbf{Q}_1 and n_2 be the number of points returned by querying \mathbf{Q}_2 . We only build the database for lower halfplanes (*i.e.* those contain $(0, -\infty)$). Note that we query all possible halfplanes that exist in the data structure, which is discretized so that the minimum distance between two is ε . The runtime is indeed linear in N and in δ^d since we only have $O((\frac{\delta}{\varepsilon})^d)$ halfplanes.

Recall that by the approximation guaranteed by the data structure, if the optimal line is contained between \mathbf{Q}_1 and \mathbf{Q}_2 , then $n_1 - n_2 \geq n_{opt}$. On the other hand, a larger number can be found elsewhere when we query some other slab, but it is guaranteed (see Figure 11) that all reported points lie within a slab of width $w' + 5\varepsilon'$. Thus we either found a slab containing the optimal digital line or we found another slab. In both cases, the number of reported points is greater than or equal to n_{opt} and the width of the slab containing them is $w' + 5\varepsilon'$.

A typical use for digital lines would be to use $w = 1$ and $0 < \varepsilon < 0.5$.

¹ The ℓ_2 distance is used in [8] while the ℓ_1 distance is used in this paper. However, the theorem still holds since the ℓ_1 distance is always greater than or equal to the ℓ_2 distance.

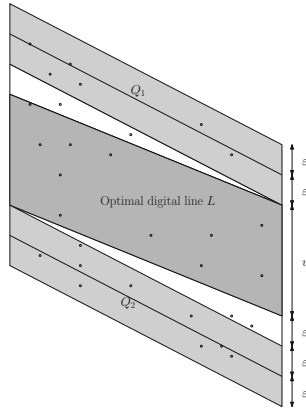


Fig. 1. The relations between optimal and approximate digital lines (see the text). Note that the approximate digital line does not **necessarily** contains the optimal line.

6 Implementation

We first note that for most practical situations, the set of points given as input is obtained through some sort of feature extraction algorithm (e.g. corner or edge detector). These extractors always require runtime which is at least linear in the size of the bounding box (in voxels). This means that our approximate algorithm is optimal in the sense of its runtime.

We implemented the algorithm described in Section 5 in C++ on a standard PC. In contrast to the optimal algorithm, the new algorithm is very simple to implement using recursion. We implemented the algorithm for $d = 3$ as this case is relatively expensive to solve with an optimal algorithm that would have a runtime of $O(N^3)$. Optimal algorithms in 3D and above are also effectively difficult to implement. Our implementation is conceptually similar in any dimension. For every box in space we first build the range counting data structure recursively. We first find the points that lie inside each one of the 8 children of the box (this is performed in logarithmic time using orthogonal range searching) and then we create the range counting data structure for each one of them recursively using the box of size ε as the smallest box. Then for building the data structure for the current box we create the set of all possible digital planes (depending on ε) and query each one of them in all children, summing the number of points.

We now go over all planes in the outer box (the bounding box of the set of input points) and for each plane, we query the number of points below this plane and below a parallel plane of distance $w + 5\varepsilon$ as described in the previous section. We simply take the pair (which is a slab) in which the number of points resulting from the subtraction of the two is the largest. A practical problem however is the memory requirement, since the algorithm has space (and roughly time) complexity $O((\delta/\varepsilon)^3)$, so in 3D it may require large amounts of memory depending on the approximation parameter.

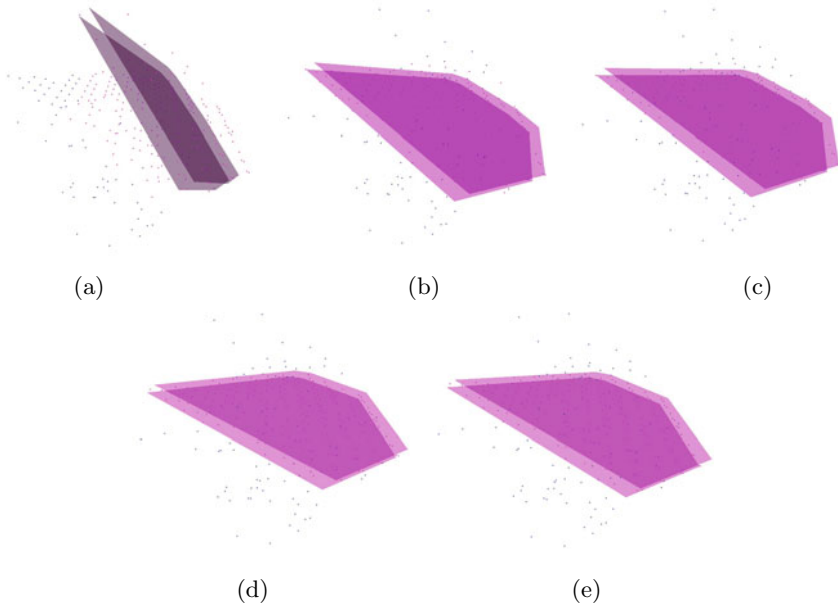


Fig. 2. Experimental results of digital plane fitting for a 3D synthetic volume data generated by 400 points in a digital plane and 100 randomly generated points (outliers). The approximation algorithm with bounded error in digital plane width is applied with the value of ε set to be 2.0 (a), 1.0 (b), 0.5 (c), and 0.25 (d): rose points are inliers and blue points are outliers. A fitted digital plane is visualized as a pair of rose parallel planes. The optimal solution (e) obtained by the exact algorithm [22] is also illustrated.

7 Experimental Results

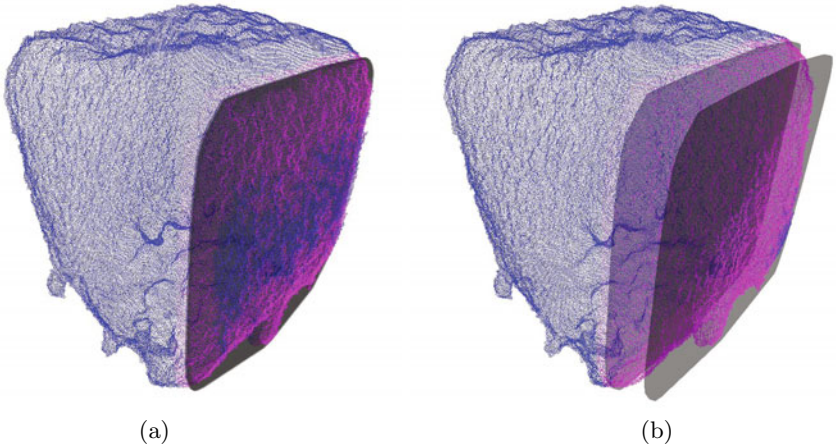
We used two 3D digital images for our experiments: some synthetic test data and a real electron nano-tomography image.

The 3D synthetic data was created such that 400 grid points were samples from a digital plane formulation with setting $w = 1$, and 100 grid points were added randomly. Four different ε values were used: 2.0, 1.0, 0.5 and 0.25. For comparison, the exact algorithm [22] with time complexity $O(N^3 \log N)$ was also applied. As seen in Table 1, the computation time is long, however it does yield the optimal solution containing 406 inlier points (*i.e.* the 400 expected points plus 6 random ones).

The approximation results indicate, as illustrated in Fig. 2 and Table 1, that the smaller the value of ε , the more precise the solution; when $\varepsilon = 2$, a solution relatively far from the optimal was obtained. Table 1 also shows that two different numbers of points were obtained for each value of ε : the first one is the number of points in a fitted digital plane with width w , and the second one is the number of points in a fitted digital plane with width $w + 5\varepsilon$. Note that the second number is guaranteed to be greater than or equal to the optimal number of inlier points by design, and it indeed converges to the optimum as ε is decreasing (see Table 1).

Table 1. The runtimes, parameters and numbers of points of fitted digital planes in Fig. 2

	runtime	parameters				nb. of points	
		a_1	a_2	a_3	a_4	w	$w + 5\epsilon$
Approximate fitting							
$\epsilon = 2.0$	31 msec	-0.0625	-0.0625	1	-10.9	110	485
$\epsilon = 1.0$	266 msec	-0.5	-0.5	1	-3.0002	300	435
$\epsilon = 0.5$	2172 msec	-0.5	-0.5	1	-3.0002	300	421
$\epsilon = 0.25$	16640 msec	-0.5625	-0.546875	1	-2.55002	362	410
Exact fitting [22]							
with exact comp.	35 min 29.109 sec	-47/81	-43/81	1	-203081/81000	406	
without exact comp.	4 min 36.908 sec	-0.580247	-0.530864	1	-2.507173		

**Fig. 3.** Results of digital plane fitting for a pre-processed 3D binary nano-tomography image containing 205001 points. The approximation algorithm with bounded error in digital plane width is applied with $\epsilon = 4$ for $w = 1$ (a) and $w = 25$ (b).

We can observe as well from Table 1 that the smaller the value of ϵ , the higher the runtime. Therefore, it is necessary in practice to find an appropriate value for ϵ , which provides a sufficiently approximate solution within a reasonable timeframe.

The second data we used was a 3D binary image generated from a electron nano-tomography image containing a cubical crystal. The image is very noisy due to the dimension of the sample and the physically-constrained tomography reconstruction method. The original image is of $512 \times 511 \times 412$ with gray values. After binarizing the image by a threshold, we detected the boundary points by using the 6-neighborhood and extracted the maximum connected component by using the 26-connectivity. Finally, we obtained 205001 points in the $512 \times 511 \times 412$ grid. This number of points is too large to apply the exact algorithm [22]: it would have required on the order of 10^{18} operations, *i.e.* many months of runtime. Instead we ran the approximate algorithm with some adjustments of

the values for ε and w . We set $\varepsilon = 4$ to obtain its runtime around 12 seconds with $w = 1$; the result is illustrated in Fig. 3(a). As we saw that the cube wall is very noisy, we also performed a fitting with $w = 25$ to obtain a thicker digital plane (see Fig. 3(b)).

8 Conclusion

In this article we have presented two approximate discrete hyperplane fitting methods with outliers. The first uses an approach based on linear programming with violations. It is continuous in nature and features interesting complexities but is difficult to implement. The second is discrete in nature, it uses an accumulation and query data structure and is easy to implement. This method features bounded error defined in this way: for a given N points, a width w and an error factor ε , the hyperplane found contains in a width equal to $w + 5\varepsilon$ at least as many points as the optimum would with a width of w . It features a computational complexity in $O(N + (\frac{\delta}{\varepsilon})^d)$, where δ is the distance between two neighbours in the hypergrid. The algorithm is therefore linear in the number of points in the set being considered, but exponential in the approximation factor with d , the geometric dimension. Memory requirements are also exponential with ε and d in the same way.

Nonetheless, as we show in the article that the exact solution is 3SUM-hard, this method is useful. The computational complexity is given in the worst case, in practice it can be much better. The algorithm is in particular well-behaved when there are few outliers. Due to lack of space, we did not show it in the paper, but the algorithm converges to the optimal solution, and in the discrete case there exists a finite ε for which the algorithm provides the optimal solution, even though this ε might be too small to be practical.

References

1. Afshani, P., Chan, T.M.: On approximate range counting and depth. *Discrete & Computational Geometry* 42(1), 3–21 (2009)
2. Aronov, B., Har-Peled, S.: On approximating the depth and related problems. *SIAM J. Comput.* 38(3), 899–921 (2008)
3. Bhowmick, P., Bhaattacharya, P.: Fast approximation of digital curves using relaxed straightness properties. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(9), 1590–1602 (2007)
4. Boyd, S., Vandenberghe, L.: *Convex optimization*. Cambridge University Press, Cambridge (2004)
5. Chum, O.: *Two-View Geometry Estimation by Random Sample and Consensus*. Ph.D. thesis, Czech Technical University, Prague, Czech Republic (July 2005)
6. Duda, R.O., Hart, P.E.: Use of the hough transformation to detect lines and curves in pictures. *Comm. ACM* 15, 11–15 (1972)
7. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24(6), 381–395 (1981)

8. da Fonseca, G.D., Mount, D.M.: Approximate range searching: The absolute model. *Comput. Geom.* 43, 434–444 (2010)
9. Gajentaan, A., Overmars, M.H.: On a class of $o(n^2)$ problems in computational geometry. *Comput. Geom.* 5, 165–185 (1995)
10. Hartley, R., Zisserman, R.: *Multiple view geometry in computer vision*. Cambridge University Press, Cambridge (2003)
11. Hough, P.V.C.: Machine analysis of bubble chamber pictures. In: *Proc. Int. Conf. High Energy Accelerators and Instrumentation* (1959)
12. Kenmochi, Y., Buzer, L., Sugimoto, A., Shimizu, I.: Discrete plane segmentation and estimation from a point cloud using local geometric patterns. *International Journal of Automation and Computing* 5(3), 246–256 (2008)
13. Kenmochi, Y., Buzer, L., Talbot, H.: Efficiently computing optimal consensus of digital line fitting. In: *International Conference on Pattern Recognition*, pp. 1064–1067 (2010)
14. Köster, K., Spann, M.: An approach to robust clustering - application to range image segmentation. *IEEE Transaction on Pattern Analysis and Machine Intelligence* 22(5), 430–444 (2000)
15. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: *Numerical recipes: the art of scientific computing*, 3rd edn. Cambridge University Press, Cambridge (2007)
16. Reveillès, J.P.: *Géométrie discrète, calcul en nombres entiers et algorithmique*. Ph.D. thesis, Thèse d'Etat, Université Louis Pasteur (1991)
17. Rousseeuw, P.J.: Least median of squares regression. *Journal of the American Statistical Association* 79(388), 871–880 (1984)
18. Shum, H.Y., Ikeuchi, K., Reddy, R.: Principal component analysis with missing data and its application to polyhedral object modeling. *IEEE Transaction on Pattern Analysis and Machine Intelligence* 17(9), 854–867 (1995)
19. Sivignon, I., Dupont, F., Chassery, J.M.: Decomposition of a three-dimensional discrete object surface into discrete plane pieces. *Algorithmica* 38, 25–43 (2004)
20. Zitova, B., Flusser, J.: Image registration methods: a survey. *Image and Vision Computing* 21, 977–1000 (2003)
21. Zrou, R., Kenmochi, Y., Talbot, H., Buzer, L., Hamam, Y., Shimizu, I., Sugimoto, A.: Optimal consensus set for digital line fitting. In: *Proceedings of IWCIA 2009, Cancun, Mexico* (2009)
22. Zrou, R., Kenmochi, Y., Talbot, H., Buzer, L., Hamam, Y., Shimizu, I., Sugimoto, A.: Optimal consensus set for digital plane fitting. In: *Proceedings of 3DIM 2009, Workshop of ICCV, Kyoto, Japan* (2009)
23. Zrou, R., Kenmochi, Y., Talbot, H., Buzer, L., Hamam, Y., Shimizu, I., Sugimoto, A.: Optimal consensus set for digital line and plane fitting. *International Journal of Imaging Systems and Technology* (2010) (in print)
24. Zrou, R., Kenmochi, Y., Talbot, H., Shimizu, I., Sugimoto, A.: Combinatorial optimization for fitting of digital line and plane. In: Wada, T., Huang, F., Lin, S. (eds.) *PSIVT 2009. LNCS*, vol. 5414. Springer, Heidelberg (2009)

Analytical Description of Digital Circles

Eric Andres¹ and Tristan Roussillon²

¹ Laboratoire XLIM, SIC Department,
University of Poitiers, BP30179, UMR CNRS 6712,
86962 Futuroscope Chasseneuil Cedex, France
`andres@sic.univ-poitiers.fr`

² Université de Lyon,
Université Lyon 2, LIRIS, UMR CNRS 5205, F-69676, France
`tristan.roussillon@liris.cnrs.fr`

Abstract. In this paper we propose an analytical description of different kinds of digital circles that appear in the literature and especially in digital circle recognition algorithms.

1 Introduction

Digital primitive recognition is an important topic for the digital geometry and pattern recognition communities. Two of the most basic primitives have been intensively studied, the digital straight line and the digital circle, and many different recognition algorithms have been proposed. One of the key elements in recognizing a digital primitive is actually knowing what object is recognized. This is not as obvious as it seems. Firstly, there are many ways of defining a digital primitive for a given Euclidean one. This means that depending on the considered definition, a same set of pixels can be recognized as a digital primitive or not. Most recognition algorithms provide parameters of the Euclidean primitive and the corresponding digital primitive is implicit. This makes any comparison between different algorithms hard because different sets are recognized. Secondly, there is the actual problem of the definition of a digital primitive. A digital primitive that is only defined as the result of an algorithm or implicitly by a set of properties does not easily allow a global mathematical description.

An analytical description of a digital primitive is interesting for several reasons: it provides a simple way of verifying whether a point or a set of points belongs to the primitive or not, it often provides generalizations that cannot be easily derived from an algorithm. For instance, the Bresenham circle has been defined for integer coordinate centers and integer radii as the result of a digitization algorithm. As we will see, its analytical description straightforwardly extends the previous definition to non integer centers and radii. Extensions to higher dimensions are also possible. Although we don't consider that in this paper, it is one of the perspectives of this paper.

J-P. Reveillès [23] proposed an analytical description for the digital straight lines in 1991. Most, if not all, the digital straight segment recognition algorithms recognize a connected subset of a kind or another of Reveillès digital straight

lines. In the different digital circle recognition algorithms, there are various digital circle definitions proposed without any analytical characterization. In this paper, we propose analytical definitions of most digital circles, *i.e.* each digital circle is defined as the solution of a system of analytical inequalities.

After a short recall on digital analytical models and the analytical description of the Andres circle, we propose an analytical description of the supercover, standard and naïve circles. In the same way, we propose an analytical description of the digital circles defined as the boundary of the Gauss digitization of Euclidean circles. We end the paper with a discussion.

2 Digital Analytical Circle Description

In this section, we propose analytical descriptions for several definitions that appear in digital circle recognition algorithms.

2.1 Recall on Digital Analytical Models

We consider here digital analytical models based on a distance d . Let us consider a Euclidean object E . The digitization $D_d(E)$ of E according to the digital analytical model associated to d is defined by:

$$D_d(E) = \left\{ p \in \mathbb{Z}^2 \mid d(p, E) \leq \frac{1}{2} \right\}.$$

This global definition is particularly interesting because several digitization models can be derived with respect to the classical distances such as the Manhattan distance d_1 , the Euclidean distance d_2 or the Tchebychev distance d_∞ . There are also some basic properties that are very useful when constructing digital objects such as, for E, F two Euclidean objects, $D_d(E \cup F) = D_d(E) \cup D_d(F)$ (see [5] for more details on the supercover analytical model).

There is an equivalent definition that involves a structuring element: the unit sphere $B_d(1)$ of diameter one for the distance d . The morphological definition can be written as follow:

$$D_d(E) = (E \oplus B_d(1)) \cap \mathbb{Z}^2$$

where $A \oplus B = \{a + b, a \in A, b \in B\}$ is the Minkowski sum.

The Euclidean region $E \oplus B_d(1)$ is called the *offset region*. The *Pythagorean* model is based on the d_2 distance, the *supercover* model is based on the d_∞ distance and the *naïve* model on the d_1 distance. These models respectively define the Andres, supercover and closed naïve digital circles.

In the following subsections, we focus on the analytical models of the Euclidean circle of center $(x_o, y_o) \in \mathbb{R}^2$ and radius $R \in \mathbb{R}^+$, denoted by $\mathcal{C}(x_o, y_o, R)$, and we derive analytical definitions of the above-mentioned digital circles from their morphological definitions.

2.2 Andres Circle

The Andres circle, which is based on the Euclidean distance, has been proposed in all dimensions by Andres [6]. In two dimensions, it is defined as follows: $(x, y) \in \mathbb{Z}^2$ belongs to the Andres circle of center (x_o, y_o) and radius R if and only if:

$$(R - \frac{1}{2})^2 \leq (x - x_o)^2 + (y - y_o)^2 < (R + \frac{1}{2})^2$$

Note that in this definition, x_o, y_o and R are not integers. If we consider a closed definition with \leq on both inequalities for the Andres circle then it is easy to see that it is associated to the distance d_2 :

$$(\mathcal{C} \oplus B_2(1)) \cap \mathbb{Z}^2$$

The recognition of Andres circles can be solved by annulus fitting, which is a problem that has been extensively studied by the computational geometry community [1]. More recently, people of the digital geometry community have also considered the problem of annulus fitting in the case of Andres circles corrupted by noise [28].

2.3 Supercover Circles

The supercover model is based on the d_∞ distance. In two dimensions, the corresponding structuring element is the unit square $B_\infty(1)$, which is the axis-aligned closed unit square, *i.e.* $B_\infty(1) = \{(x, y) \in \mathbb{R}^2 \mid \max(|x|, |y|) \leq \frac{1}{2}\}$.

The supercover model is very well adapted for linear objects and every linear object can be described analytically in this model [5]. However, it can also be applied on pieces of \mathcal{C}^2 curves where the slope of the tangent monotonously increases or decreases like in circle quadrants. For instance, let us consider the circular arc \mathcal{A} of the circle $\mathcal{C}(0, 0, R)$ between the angles 0 to $\frac{\pi}{2}$. The offset region $\mathcal{A} \oplus B_\infty(1)$ is the union between the closed unit squares centered on $(0, R)$ and $(R, 0)$ at both ends of \mathcal{A} , and the region bounded by the translation of \mathcal{A} by the vector $(\frac{1}{2}, \frac{1}{2})$, the translation of \mathcal{A} by the vector $(-\frac{1}{2}, -\frac{1}{2})$, the straight segment joining $(-\frac{1}{2}, R - \frac{1}{2})$ and $(\frac{1}{2}, R + \frac{1}{2})$, the straight segment joining $(R - \frac{1}{2}, -\frac{1}{2})$ and $(R + \frac{1}{2}, \frac{1}{2})$ (fig. 1a).

Due to symmetries, the offset region of the three other quadrants is defined in the same way so that each point of $\mathcal{C} \oplus B_\infty(1)$ either lies in one of the four closed unit squares centered in $(0, R)$, $(R, 0)$, $(0, -R)$, $(-R, 0)$ or belongs to one of the four disks of radius R and center $(-\frac{1}{2}, -\frac{1}{2})$, $(\frac{1}{2}, \frac{1}{2})$, $(-\frac{1}{2}, \frac{1}{2})$, $(\frac{1}{2}, -\frac{1}{2})$, but not to both of them (fig. 1b).

We can thus derive an analytical description of the supercover circle:

Proposition 1 (Supercover of a circle). *A point $(x, y) \in \mathbb{Z}^2$ belongs to the supercover circle $\mathbb{C}_\infty(x_o, y_o, R) = ((\mathcal{C} \oplus B_\infty(1)) \cap \mathbb{Z}^2)$, if and only if:*

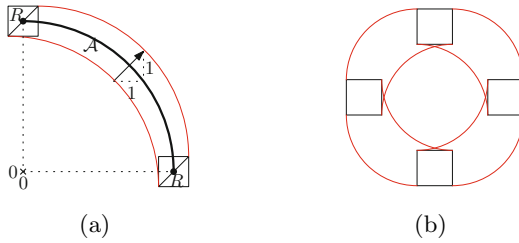


Fig. 1. Construction of the offset region used to define the supercover of a circle

$$\begin{aligned}
 &|y - y_o| \leq \frac{1}{2} \text{ and } \left| (|x - x_o| - R) \right| \leq \frac{1}{2} \\
 &\quad \text{or} \\
 &|x - x_o| \leq \frac{1}{2} \text{ and } \left| (|y - y_o| - R) \right| \leq \frac{1}{2} \\
 &\quad \text{or} \\
 &R^2 - \frac{1}{2} - (|x - x_o| + |y - y_o|) \leq (x - x_o)^2 + (y - y_o)^2 \leq \\
 &\quad R^2 - \frac{1}{2} + (|x - x_o| + |y - y_o|)
 \end{aligned}$$

Proof. The first part of the analytical description corresponds to the squares at the cardinal points and is obvious. We get the last part of the analytical description by developing and applying axial symmetries to the equations for the first quadrant $(x - x_o - \frac{1}{2})^2 + (y - y_o - \frac{1}{2})^2 \geq R^2$ and $(x - x_o + \frac{1}{2})^2 + (y - y_o + \frac{1}{2})^2 \leq R^2$. □

The supercover of $\mathbb{C}_\infty(0, 0, 5)$ and $\mathbb{C}_\infty(\frac{1}{2}, \frac{1}{2}, 5)$ are respectively depicted in fig. 2a and b.

Note that Lincke proposed another interpretation of this result based on mathematical morphology operations [17] [fig. 4].

Note in addition that Nakamura and Aizawa, based on a cellular scheme, defined a digital disk [20] that is actually a supercover disk. The outer border of their digital disk is thus also the outer border of a supercover circle.

2.4 Standard Analytical Circles

The standard model has been defined only for linear primitives in [4]. In the supercover model, when the Euclidean object passes through a point p composed only of half-integer coordinates, there is a *bubble*, i.e. four digital points that are the vertices of the axis-aligned closed unit square centered on p . For instance, the supercover of the point $(\frac{1}{2}, \frac{1}{2})$ is composed of the digital points $\{(0, 0); (0, 1); (1, 0); (1, 1)\}$. Similarly, in fig. 2b, the points $(5 + \frac{1}{2}, \frac{1}{2}), (4 + \frac{1}{2}, 3 + \frac{1}{2}), (3 + \frac{1}{2}, 4 + \frac{1}{2}), (\frac{1}{2}, 5 + \frac{1}{2})$ (only for the first quadrant) are lying on $\mathcal{C}(\frac{1}{2}, \frac{1}{2}, 5)$ and are thus center of bubbles.

There can be two different definitions of standard circles depending if we remove the outer or inner border of the offset region $\mathcal{C} \oplus B_\infty(1)$. The analytical

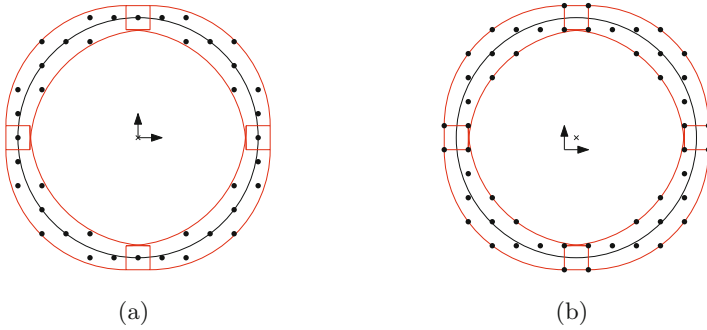


Fig. 2. The supercover of $\mathbb{C}_\infty(0, 0, 5)$ (a) and $\mathbb{C}_\infty(\frac{1}{2}, \frac{1}{2}, 5)$ (b) is depicted with black disks

description of the outer standard circle $\mathbb{C}_\infty^+(x_o, y_o, R)$ is obtained by replacing $|x - x_o| \leq R + \frac{1}{2}$ by $|x - x_o| < R + \frac{1}{2}$, $|y - y_o| \leq R + \frac{1}{2}$ by $|y - y_o| < R + \frac{1}{2}$ and $(x - x_o)^2 + (y - y_o)^2 \leq R^2 - \frac{1}{2} + (|x - x_o| + |y - y_o|)$ by $(x - x_o)^2 + (y - y_o)^2 < R^2 - \frac{1}{2} + (|x - x_o| + |y - y_o|)$ in the analytical description of the supercover circle (the outer border is removed). Similarly, the analytical description of the inner standard circle $\mathbb{C}_\infty^-(x_o, y_o, R)$ is obtained by simply replacing $R^2 - \frac{1}{2} - (|x - x_o| + |y - y_o|) \leq (x - x_o)^2 + (y - y_o)^2$ by $R^2 - \frac{1}{2} - (|x - x_o| + |y - y_o|) < (x - x_o)^2 + (y - y_o)^2$ in the supercover circle definition (the inner border is removed). The inner standard circle $\mathbb{C}_\infty^-(\frac{1}{2}, \frac{1}{2}, 5)$ is depicted in fig. 3.a.

Most of the bubbles are removed in the standard model (compare for instance fig. 2.b and fig. 3.a). However, neither the outer nor the inner standard circles are always *simply* 4-connected because it may remain a bubble around one of the four cardinal points (like in fig. 3.a). The inner standard circle is however interesting

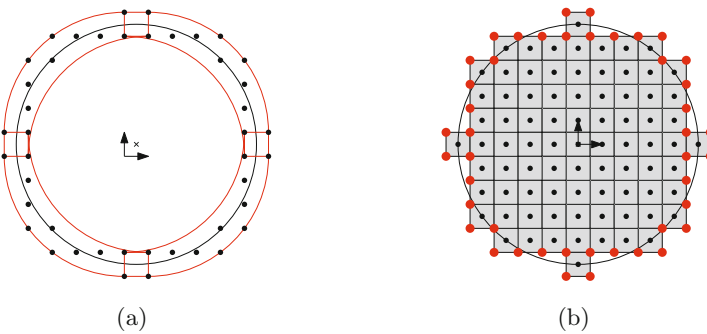


Fig. 3. The inner standard analytical circle $\mathbb{C}_\infty^-(\frac{1}{2}, \frac{1}{2}, 5)$ is depicted in (a) with black disks. The Kovalevsky circle $\mathbb{K}(0, 0, 5)$ is depicted in (b) with big disks along the boundary of the gray area. Note that $\mathbb{C}_\infty^-(\frac{1}{2}, \frac{1}{2}, 5)$ and $\mathbb{K}(0, 0, 5)$ are similar objects up to a translation

because it corresponds to the Kovalevsky circle [15], denoted by $\mathbb{K}(x_o, y_o, R)$ and defined as the set of points of half-integer coordinates $(x, y) \in (\mathbb{Z} + \frac{1}{2}) \times (\mathbb{Z} + \frac{1}{2})$ that belongs to the boundary of the dilatation of the Gauss digitization of the interior of $\mathcal{C}(x_o, y_o, R)$ by $B_\infty(1)$, i.e. the boundary of $\{(i, j) \in \mathbb{Z}^2 \mid (i - x_o)^2 + (j - y_o)^2 \leq R^2\} \oplus B_\infty(1)$.

In fig. 3b, the Gauss digitization of $\mathcal{C}(0, 0, 5)$ is depicted with black disks. The gray area is its dilatation by $B_\infty(1)$. Kovalevsky circle is depicted with red disks lying along the boundary of the gray area.

Proposition 2. *The point $(x, y) \in \mathbb{Z}^2$ belong to $\mathbb{C}_\infty^-(x_o + \frac{1}{2}, y_o + \frac{1}{2}, R)$ if and only if the point $(x - \frac{1}{2}, y - \frac{1}{2})$ belongs to $\mathbb{K}(x_o, y_o, R)$.*

Proof. Firstly, we will prove that if the point $(x - \frac{1}{2}, y - \frac{1}{2})$ belong to $\mathbb{K}(x_o, y_o, R)$, then the point (x, y) belongs to $\mathbb{C}_\infty^-(x_o + \frac{1}{2}, y_o + \frac{1}{2}, R)$ (i) and secondly, we will prove that if the point $(x - \frac{1}{2}, y - \frac{1}{2})$ does not belong to $\mathbb{K}(x_o, y_o, R)$, then (x, y) does not belong to $\mathbb{C}_\infty^-(x_o + \frac{1}{2}, y_o + \frac{1}{2}, R)$.

Let us shortly denote by \mathcal{C} the circle of center $(x_o, y_o) \in \mathbb{R}^2$ and radius $R \in \mathbb{R}^+$. For all points $p(x, y) \in (\mathbb{Z} + \frac{1}{2}) \times (\mathbb{Z} + \frac{1}{2})$, the four vertices of $p \oplus B_\infty(1)$ are points of \mathbb{Z}^2 . Those that are enclosed by \mathcal{C} or lie on \mathcal{C} are referred as *foreground points*, the others are referred as *background points*.

(i) For all points $p(x - \frac{1}{2}, y - \frac{1}{2}) \in \mathbb{K}(x_o, y_o, R)$, at least one vertex of $p \oplus B_\infty(1)$ is a foreground point and at least one is a background point, because p is assumed to belong to $\mathbb{K}(x_o, y_o, R)$. As a consequence, \mathcal{C} must intersect $p \oplus B_\infty(1)$ and by duality $\mathcal{C} \oplus B_\infty(1)$ contains p . Note that since \mathcal{C} cannot pass through any background point, p does not belong to the inner border of $\mathcal{C} \oplus B_\infty(1)$ (if any), but is either in its interior or is lying on the outer border. Since this membership is preserved under translation, (x, y) belongs to $\mathbb{C}_\infty^-(x_o + \frac{1}{2}, y_o + \frac{1}{2}, R)$.

(ii) For all points $p(x - \frac{1}{2}, y - \frac{1}{2}) \notin \mathbb{K}(x_o, y_o, R)$, the four vertices of $p \oplus B_\infty(1)$ are either all foreground points or all background points. In the last case, \mathcal{C} does not intersect $p \oplus B_\infty(1)$ and by duality $\mathcal{C} \oplus B_\infty(1)$ does not contain p and we are done. In the first case, \mathcal{C} may pass through one of the foreground points and p may thus lie in the inner border of $\mathcal{C} \oplus B_\infty(1)$ (which always exists in this case). Since the inner border of the offset region is removed in the standard model and since the incident relations are preserved under translation, (x, y) does not belong to $\mathbb{C}_\infty^-(x_o + \frac{1}{2}, y_o + \frac{1}{2}, R)$. □

Due to this proposition, the recognition algorithm of Kovalevsky [15] provides a way of recognizing inner standard circles.

2.5 Closed Naïve and Naïve Analytical Circles

The naïve model, introduced in [3], is based on the d_1 distance. In two dimensions, the corresponding structuring element is the unit square $B_1(1)$, which is a square of side size $\frac{\sqrt{2}}{2}$ and with a 45° rotation compared to $B_\infty(1)$, i.e. $B_1(1) = \{(x, y) \in \mathbb{R}^2 \mid (|x| + |y|) \leq \frac{1}{2}\}$.

The analytical description of the closed naïve circle is therefore very similar to the one of the supercover circle (compare the offset region defining a closed naïve circle in fig. 4a and the one defining a supercover circle in fig. 1b).

Proposition 3 (Closed naïve circle). *A point $(x, y) \in \mathbb{Z}^2$ belongs to the closed naïve circle $\mathbb{C}_1(x_o, y_o, R) = \left(\mathcal{C} \oplus B_1(1) \right) \cap \mathbb{Z}^2$, if and only if:*

$$\begin{aligned} & |(x - y) - (x_o - y_o)| \leq \frac{1}{2} \text{ and } \left| |x + y - (x_o + y_o)| - R\sqrt{2} \right| \leq \frac{1}{2} \\ & \qquad \qquad \qquad \text{or} \\ & |(x + y) - (x_o + y_o)| \leq \frac{1}{2} \text{ and } \left| |x - y - (x_o - y_o)| - R\sqrt{2} \right| \leq \frac{1}{2} \\ & \qquad \qquad \qquad \text{or} \\ & R^2 - \frac{1}{4} - \max(|x - x_o|, |y - y_o|) \leq (x - x_o)^2 + (y - y_o)^2 \\ & \qquad \qquad \qquad \leq R^2 - \frac{1}{4} + \max(|x - x_o|, |y - y_o|) \end{aligned}$$

Proof. The proof of this proposition is very similar to the proof of proposition 1. The last part of the equations is obtained by developing and applying the corresponding symmetries to $(x - x_o - \frac{1}{2})^2 + (y - y_o)^2 \geq R^2$ and $(x - x_o + \frac{1}{2})^2 + (y - y_o)^2 \leq R^2$. □

Similarly to the two definitions of standard circles, it is possible to define an inner naïve circle, denoted by $\mathbb{C}_1^-(x_o, y_o, R)$, and an outer naïve circle, denoted by $\mathbb{C}_1^+(x_o, y_o, R)$, by removing the inner or outer border of the offset region $\mathcal{C} \oplus B_1(1)$. Due to this convention, when \mathcal{C} crosses a point of coordinates $(x + \frac{1}{2}, y)$ in the first octant, \mathbb{C}_1^- does not contain (x, y) , whereas \mathbb{C}_1^+ does not contain $(x + 1, y)$. For instance, in fig. 4b, we can see the closed naïve circle $\mathbb{C}_1(\frac{1}{4}, \frac{1}{4}, \frac{7}{2})$. The Euclidean circle \mathcal{C} crosses the points of coordinates $(-1, 3 + \frac{1}{2})$, $(3 + \frac{1}{2}, -1)$, $(-2, -2 - \frac{1}{2})$ and $(-2 - \frac{1}{2}, -2)$ (clockwise from the top-left point), which implies that the points $(-2, 3)$, $(0, 4)$, $(4, 0)$, $(3, -2)$, $(-2, -3)$, $(-2, -2)$ and $(-3, -2)$ have three 8-neighbors. However, $\mathbb{C}_1^-(\frac{1}{4}, \frac{1}{4}, \frac{7}{2})$ (resp. $\mathbb{C}_1^+(\frac{1}{4}, \frac{1}{4}, \frac{7}{2})$) is simply 8-connected because it does not contain the points $(-1, 3)$, $(3, -1)$, $(-2, -2)$ (resp. $(-1, 4)$, $(4, -1)$, $(-2, -3)$ and $(-3, -2)$).

An inner or outer naïve circle may nonetheless not always be simply 8-connected because sharp corners may occur at octant boundaries (fig. 4c). Exactly the same thing happens for Bresenham circles 7 as it is well known (see for instance 18 [section 5]) because of the following proposition:

Proposition 4 (Bresenham Circle). *Bresenham circle is a closed, inner and outer naïve circle.*

Proof. Let us assume that x_o, y_o, R are integers. A Bresenham circle is a closed 8-connected digital curve that is the digitization of $\mathcal{C}(x_o, y_o, R)$. Moreover, its points are the closest ones to $\mathcal{C}(x_o, y_o, R)$ 16. As such it corresponds to the naïve digitization model.

Moreover no point $(x \pm \frac{1}{2}, y)$ or $(x, y \pm \frac{1}{2})$, with $(x, y) \in \mathbb{Z}^2$, belongs to a Euclidean circle that has a center of integer coordinates and an integer radius.

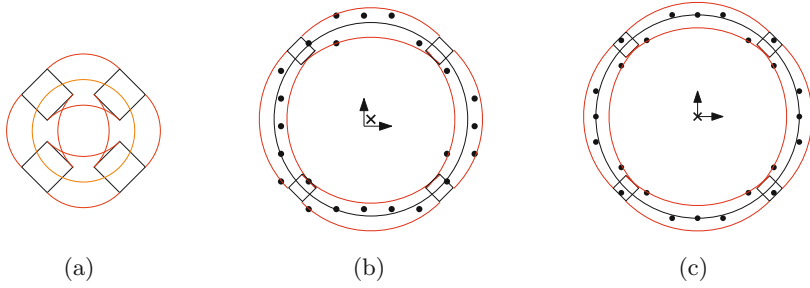


Fig. 4. Offset region defining a closed naive circle (a). A closed naive circle $\mathbb{C}_1(\frac{1}{4}, \frac{1}{4}, \frac{7}{2})$ (b). Bresenham circle of radius 4 that is not simply 8-connected (c).

The closed, inner and outer naive models of such circles are thus identical and corresponds to their Bresenham digitization. \square

This proposition shows that the analytical description we propose for naive circles is an extension of the Bresenham circles to arbitrary centers and radii. If we consider that the natural extension of the Bresenham circle corresponds to the circles that are simply 8-connected except around octant boundaries, then the outer naive circle is the best choice. This corresponds to the definition proposed by Pham [22] (see [2]). Note that there is a slight mistake in the starting point in Pham’s generation algorithm. This type of mistake is difficult to spot with a generation algorithm. One advantage of an analytical definition is that it can be used to test such algorithms.

Several papers have dealt with the problem of recognizing naive circles: the paper of Pham [22] in 1992 for outer naive circles and the papers of Sauer [24] in 1993 and Damaschke [9] in 1995 for Bresenham circles.

3 Gauss Type Digitized Circles

In this section, we propose two analytical definitions that do not directly rely on the global model presented in section 2.1 but that are very similar.

Definition 1 (d_1 -Gauss circle). A point (x, y) belongs to the d_1 -Gauss circle $\mathbb{G}_\infty(x_o, y_o, R)$ if and only if:

$$R^2 - 2 \max(|x - x_o|, |y - y_o|) - 1 < (x - x_o)^2 + (y - y_o)^2 \leq R^2$$

Definition 2 (d_∞ -Gauss circle). A point (x, y) belongs to the d_∞ -Gauss circle $\mathbb{G}_1(x_o, y_o, R)$ if and only if:

$$R^2 - 2(|x - x_o| + |y - y_o|) - 1 < (x - x_o)^2 + (y - y_o)^2 \leq R^2$$

It is easy to sketch the region where the points of $\mathbb{G}_1(x_o, y_o, R)$ or $\mathbb{G}_\infty(x_o, y_o, R)$ lie (it is bounded by circular arcs in fig. 5).

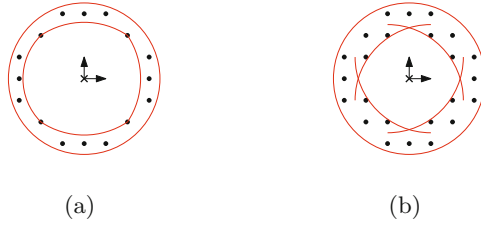


Fig. 5. Illustration of $\mathbb{G}_1(0,0,3.5)$ in (a) and $\mathbb{G}_\infty(0,0,3.5)$ in (b)

These digital circles actually correspond to the boundary of the Gauss digitization of \mathcal{C} . More precisely:

Proposition 5. *The d_1 -Gauss (resp. d_∞ -Gauss) circle $\mathbb{G}_1(x_o, y_o, R)$ (resp. $\mathbb{G}_\infty(x_o, y_o, R)$) is the set of points of the Gauss digitization of the interior of $\mathcal{C}(x_o, y_o, R)$, denoted by $D_G(\mathcal{C})$, such that their 4-(resp. 8-)neighborhood is not totally included in $D_G(\mathcal{C})$.*

Proof. We will focus on $\mathbb{G}_1(x_o, y_o, R)$ because the proof about $\mathbb{G}_\infty(x_o, y_o, R)$ is the same. Firstly (i), we will prove that if $p \in \mathbb{G}_1(x_o, y_o, R)$, then the 4-neighborhood of p is not totally included in $D_G(\mathcal{C})$ and secondly (ii), we will prove that if $p \notin \mathbb{G}_1(x_o, y_o, R)$, then either the 4-neighborhood of p is totally included in $D_G(\mathcal{C})$ or not included at all.

(i) For all $p(x, y) \in \mathbb{G}_1(x_o, y_o, R)$, $(x - x_o)^2 + (y - y_o)^2 \leq R^2$ and therefore $p \in D_G(\mathcal{C})$. Let us now assume that p lies in the first octant, i.e. $(x - x_o) > (y - y_o) \geq 0$. On the one hand $(x + 1 - x_o)^2 + (y - y_o)^2 = (x - x_o)^2 + (y - y_o)^2 + 2(x - x_o) + 1$. On the other hand $(x - x_o)^2 + (y - y_o)^2 > R^2 - 2(x - x_o) - 1$ due to definition \square . As a result, $(x + 1 - x_o)^2 + (y - y_o)^2 > R^2$, i.e. $(x + 1, y) \notin D_G(\mathcal{C})$. Due to symmetries, we can conclude that the 4-neighborhood of p is not totally included in $D_G(\mathcal{C})$ for all $p(x, y) \in \mathbb{G}_1(x_o, y_o, R)$.

(ii) For all $p(x, y) \notin \mathbb{G}_1(x_o, y_o, R)$, two cases must be distinguished. If $p \notin D_G(\mathcal{C})$, we are done. Otherwise, let us assume that p lies in the first octant, i.e. $(x - x_o) > (y - y_o) \geq 0$. Since $(x - x_o)^2 + (y - y_o)^2 \leq R^2 - 2(x - x_o) - 1$ in that case, we have $(x + 1 - x_o)^2 + (y - y_o)^2 < R^2$, i.e. $(x + 1, y) \in D_G(\mathcal{C})$. Due to symmetries, we can conclude that the 4-neighborhood of p is totally included in $D_G(\mathcal{C})$ in that case and we are done. \square

The d_1 -Gauss circles, also referred as circles digitized under Kim scheme in [20,12,22], appear in many different recognition algorithms [13,14,21,12,10,8].

4 Discussion and Perspectives

In this paper we have presented analytical inequalities describing the supercover, inner and outer standard, closed naïve, inner and outer naïve, d_1 - and d_∞ -Gauss circles.

Fiorio *et. al.* [11] proposed an analytical characterization for standard and naïve circles that is very close to the one we are proposing here. They obtained their formula based on differential considerations and thus obtained only the last part of the analytical descriptions. Geometrically, the offset region involved in their approach is only made up with four disks without any square. In addition, the Euclidean circle whose parameters are the same as the ones of the standard or naïve circle is not perfectly centered within the offset region. Therefore, their approach could not be used to characterize existing digital circles (like Kovalevsky circle) as we did in this paper.

Having an analytical characterization has many advantages: it provides a way of verifying if a given set of digital points is a given digital circle or a subset of such a digital circle, it provides a way of verifying the correctness of digital circle generation algorithms. Furthermore, our approach leads to a unified framework for digital circle recognition algorithms based on linear programming techniques. The only exception is Andres circles that can be handled through algorithms based on annulus fitting.

Let Σ be a set of n digital points. Is Σ a given digital circle $\mathbb{C}(x_o, y_o, R)$ (where $\mathbb{C} \in \{\mathbb{C}_\infty, \mathbb{C}_\infty^-, \mathbb{C}_\infty^+, \mathbb{C}_1, \mathbb{C}_1^-, \mathbb{C}_1^+, \mathbb{G}_\infty, \mathbb{G}_1\}$) ?

We give below a general scheme in two steps in order to solve the recognition problem using linear programming. This approach is not new and has been used in [9] for Bresenham circles, but we extend it to all the above-mentioned digital circles.

The first step consists in setting a straight segment joining a point s and a point t to each digital point $p \in \Sigma$, such that $\mathbb{C}(x_o, y_o, R)$ intersects $[st]$ if and only if p belongs to $\mathbb{C}(x_o, y_o, R)$. This step can be easily performed if the octant (with respect to (x_o, y_o)) where each $p \in \Sigma$ lies is known. We do not provide further details due to lack of space, but the octant of all $p \in \Sigma$ can be deduced from Σ in linear time. The straight segments assigned to each digital point of $\mathbb{C}_\infty(0, 0, 5)$ (a), $\mathbb{C}_1(0.5, 0.5, 3.5)$ (b) and $\mathbb{G}_1(0, 0, 3)$ (c) are depicted in fig. 6. Note that s is not included for $\mathbb{C} \in \{\mathbb{C}_\infty^+, \mathbb{C}_1^+\}$ in $[st]$, whereas t is not included in $[st]$ for $\mathbb{C} \in \{\mathbb{C}_\infty^-, \mathbb{C}_1^-, \mathbb{G}_\infty, \mathbb{G}_1\}$ (see fig. 6c for instance).

In the second step, the sets of points $s \in S$ and $t \in T$ provide the constraints of a convex program that be translated into a linear one [21,9] and can then be solved in linear-time using Megiddo prune and search technique [19] or in expected linear-time using Seidel randomized technique [25].

That's why all the recognition algorithms that appear in the literature [13,14,21,12,15,22,24,27,10,8] are all techniques of solving a unique linear program. The difference lies in the manner of solving the problem.

In [21,9], a 3D point belonging to the intersection of $2n$ half-spaces in the parameters space is searched with Megiddo algorithm. In the space that is dual to the parameters space, a plane separating two sets of n 3D points is searched using tools coming from computational geometry [10]. In [12,15,22,27,8] a 2D point belonging to the intersection of n^2 half-planes is searched in the original plane using either brute-force algorithms [15,22,27] or tools coming from computational geometry [12,8].

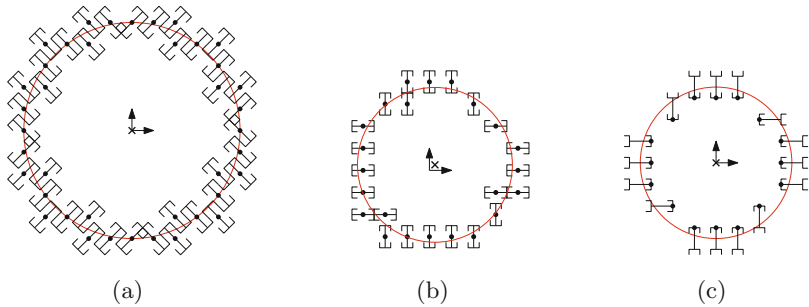


Fig. 6. Constraints assigned to each digital point of $\mathbb{C}_\infty(0, 0, 5)$ (a), $\mathbb{C}_1(0.5, 0.5, 3.5)$ (b) and $\mathbb{G}_1(0, 0, 3)$ (c)

One of the main perspective of this paper is of course the extensions that analytical descriptions allow: extension to thick digital circles (by considering structuring elements $B_d(k)$ with $k > 1$) and extension to higher dimensions, which seems possible but not trivial. Another perspective is the extension to more complex algebraic curves [26].

References

1. Agarwal, P.K., Sharir, M.: Efficient randomized algorithms for some geometric optimization problems. In: Proceedings of the 11th Annual ACM Symposium on Computational Geometry, pp. 326–335 (1995)
2. Andres, E.: Discrete circles, rings and spheres. *Computer and Graphics* 18(5), 695–706 (1994)
3. Andres, E.: Modélisation Analytique Discrète d’Objets Géométriques. Habilitation à diriger des recherches, Université de Poitiers (2000)
4. Andres, E.: Discrete linear objects in dimension n: the standard model. *Graphical Models* 65(1-3), 92–111 (2003)
5. Andres, E.: The supercover of an m-flat is a discrete analytical object. *TCS* 406(1-2), 8–14 (2008), <http://www.sciencedirect.com/science/article/B6V1G-4T4J88B-2/2/4101ff67ac5a1fb9c8028c53147a5218>, discrete Tomography and Digital Geometry: In memory of Attila Kuba
6. Andres, E., Jacob, M.A.: The discrete analytical hyperspheres. *IEEE Transactions on Visualization and Computer Graphics* 3(1), 75–86 (1997)
7. Bresenham, J.: A linear algorithm for incremental digital display of circular arcs. *Communications of the ACM* 20(2), 100–106 (1977)
8. Coeurjolly, D., Gérard, Y., Reveillès, J.P., Tougne, L.: An Elementary Algorithm for Digital Arc Segmentation. *Discrete Applied Math.* 139(1-3), 31–50 (2004)
9. Damaschke, P.: The Linear Time Recognition of Digital Arcs. *Pattern Recognition Letters* 16, 543–548 (1995)
10. Efrat, A., Gotsman, C.: Subpixel Image Registration Using Circular Fiducials. *Int. Journal of Computational Geometry & Applications* 4(4), 403–422 (1994)
11. Fiorio, C., Jamet, D., Toutant, J.L.: Discrete circles: an arithmetical approach with non-constant thickness. In: *Vision Geometry XIV, Electronic Imaging, SPIE*, vol. 6066, pp. 60660C.1–60660C.12 (2006)

12. Fisk, S.: Separating Points Sets by Circles, and the Recognition of Digital Disks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8, 554–556 (1986)
13. Kim, C.E.: Digital Disks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6(3), 372–374 (1984)
14. Kim, C.E., Anderson, T.A.: Digital Disks and a Digital Compactness Measure. In: *Annual ACM Symposium on Theory of Computing*, pp. 117–124 (1984)
15. Kovalevsky, V.A.: New Definition and Fast Recognition of Digital Straight Segments and Arcs. In: *International Conference on Pattern Analysis and Machine Intelligence*, pp. 31–34 (1990)
16. Kulpa, Z., Doros, M.: Freeman digitization of integer circles minimizes the radial error. *Computer Graphics and Image Processing* 17(2), 181–184 (1981)
17. Lincke, C., Wüthrich, C.A.: Towards a unified approach between digitization of linear objects and discrete analytical objects. In: *WSGG 2000*, University of West Bohemia, Plzen Tcheque Republic, pp. 124–131 (2000)
18. McIlroy, M.D.: Best approximate circles on integer grids. *ACM Transactions on Graphics* 2(4), 237–263 (1983)
19. Megiddo, N.: Linear Programming in Linear Time When the Dimension Is Fixed. *SIAM Journal on Computing* 31, 114–127 (1984)
20. Nakamura, A., Aizawa, K.: Digital Circles. *Computer Vision, Graphics, and Image Processing* 26(2), 242–255 (1984)
21. O'Rourke, J., Kosaraju, S.R., Megiddo, N.: Computing Circular Separability. *Discrete and Computational Geometry* 1, 105–113 (1986)
22. Pham, S.: Digital Circles With Non-Lattice Point Centers. *The Visual Computer* 9, 1–24 (1992)
23. Reveillès, J.P.: *Géométrie Discrète, calculs en nombres entiers et algorithmique*. Thèse d'état, Université Louis Pasteur (1991)
24. Sauer, P.: On the Recognition of Digital Circles in Linear Time. *Computational Geometry* 2(5), 287–302 (1993)
25. Seidel, R.: Small-dimensional linear programming and convex hulls made easy. *Discrete and Computational Geometry* 6(1), 423–434 (1991)
26. Tajine, M., Ronse, C.: Hausdorff discretizations of algebraic sets and diophantine sets. In: Nyström, I., Sanniti di Baja, G., Borgefors, G. (eds.) *DGCI 2000*. LNCS, vol. 1953, pp. 99–110. Springer, Heidelberg (2000), <http://portal.acm.org/citation.cfm?id=648319.754197>
27. Worring, M., Smeulders, A.W.M.: Digitized Circular Arcs: Characterization and Parameter Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17(6), 554–556 (1995)
28. Zrour, R., Largeteau-Skapin, G., Andres, E.: Optimal Consensus set for Annulus Fitting. In: Debled-Rennesson, I., et al. (eds.) *DGCI 2011*. LNCS, vol. 6607, pp. 238–249. Springer, Heidelberg (2011)

Circular Arc Reconstruction of Digital Contours with Chosen Hausdorff Error

Bertrand Kerautret^{1,2}, Jacques-Olivier Lachaud²,
and Thanh Phuong Nguyen¹

¹ LORIA (UMR CNRS 7503) Nancy University, France
{kerautret,nguyentp}@loria.fr

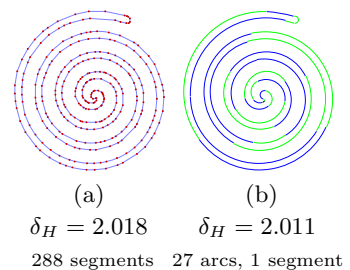
² LAMA (UMR CNRS 5127), University of Savoie, France
jacques-olivier.lachaud@univ-savoie.fr

Abstract. We address the problem of constructing an approximate continuous representation of a digital contour with guarantees on the Hausdorff error between the digital shape and its reconstruction. Instead of polygonalizing the contour, we propose to reconstruct the shape with circular arcs. To do so, we exploit the recent curvature estimators. From their curvature field, we introduce a new simple and efficient algorithm to approximate a digital shape with as few arcs as possible at a given scale, specified by a maximal admissible Hausdorff distance. We show the potential of our reconstruction method with numerous experiments and we also compare our results with some recent promising approaches. Last, all these algorithms are available online for comparisons on arbitrary shapes.

1 Introduction

Digital curve representation or approximation by simple primitives is useful for further shape analysis (recognition, matching, etc). Many methods approximate digital curves by a polygonal contour, using corner points or multi-scale analysis [6]. We propose here to work with higher order primitives like arcs of circle. This representation is more efficient for curved shapes like the one illustrated on the following floating figure and captures better its geometry.

In a previous work [11] we have explored the potential of different curvature estimators for corner detection. Together with a scale parameter, the obtained polygonal reconstructions were able to represent the contour by adapting the number of points according to the local value of curvature (see for example the polygon (a) of the upper floating figure). However as shown in figure (b), the circle arc primitive is much more efficient to represent numerous shapes. The previous example shows how arcs based representation gives a more



compact description for the same accuracy (same Hausdorff error δ_H , more faithful normals). In this paper we investigate curvature based arc reconstruction by introducing a simple reconstruction algorithm which exploits different recent curvature estimators.

Among previous works about reconstruction of digital contours with higher order primitives, we can mention the work of Rosin *et al.* [16], who constructed firstly a polygonal description and detected fitting arcs by grouping connected lines. Horng *et al.* [8] and Tortorella [18] introduced curve-fitting methods with an approach based on dynamic programming. Bodansky [4] presented a method for the approximation of a polyline with straight segments, circular arcs and free curves. It contains two steps. The first step is the segmentation of polygonal lines into fragments (short polygonal lines) and the second step is the approximation of the fragments by geometric primitives. If some fragments can not be approximated by geometric primitives with acceptable precision, they are recognized as free curves.

It is obvious that curvature information is meaningful for curve reconstruction by circle arcs and segments. A circle arc corresponds to a constant part in the curvature profile of the studied curve. Some methods have exploited this measure for curve reconstruction. Chen *et al.* [2] proposed a method for segmenting a digital curve into lines and arcs from curvature profile in which the number of primitives is given. This procedure contains two stages. The first stage computes a starting set of break points and determines an initial approximation by arcs and lines based on this set. This stage relies on the detection of significant changes along the curvature profile. The second stage is an optimization phase which adjusts the break points until the fitting error is locally minimized. Afterwards, Horng [7] proposed an adaptive smoothing approach for decomposition of a digital curve into arcs and lines. The input curve is segmented into arcs and lines according to the smoothed curvature representation. The curvature profile is determined by Gaussian filtering. Then, it is smoothed with an adaptive smoothing technique. Similarly, Salmon *et al.* [17] presented a method for decomposing a curve into arcs and segments according to the curvature profile too. They used a notion of discrete curvature which is related to the circumscribed circle induced by blurred segments. Their main idea is to extract key points on the curvature profile, which are then used for reconstruction. However the instability of this curvature estimator induces some complex curvature processing which adds new parameters and reduce the applicability of the approach.

We propose here to examine the potentiality of three recent curvature estimators in the context of reconstruction with circle arcs. Their main properties (stability wrt noise especially) are briefly described in Section 2. The reconstruction algorithm is presented in Section 3, and uses indifferently two of these estimators. We finally validate our reconstruction technique with several experimentations in Section 4. The visual curvature reconstruction [13] and the method of [15] act as reference reconstruction methods.

2 Recent Curvature Estimators

An overview of different curvature estimators are given in the following and more details can be found in their respective reference (see also the comparative evaluation in [11]).

Global Minimization Curvature (GMC). Curvature estimation is very problematic since infinitely many shapes have the same digitization. The idea of the GMC estimator [9] is to determine, among all possible shapes that have the same digitization as the digital object under study, the shape which minimizes its squared curvature. This shape is the most probable or expected shape if only a digital object is given, since it is the smoothest possible. Then the curvature field estimation is simply the curvature field of the minimal shape.

It is not a trivial task to determine this minimal shape. A phase-field approach is proposed in [1]. In [9], the proposed approximate optimization method provides a piecewise constant curvature field, so as the reconstructed shape boundary is made of circular arcs with tangent continuity. Furthermore it takes into account possible noise in the input data, with the use of blurred segments as a preprocess [3]. This estimator determines a curvature field with the smallest possible number of inflexion points. It is also almost rotation invariant due to processing with maximal digital straight segments. Its stability makes it particularly suitable to our reconstruction algorithm.

Binomial Convolution Curvature (BCC). This estimator was proposed by Malgouyres *et al.* [14,5] as a discrete alternative to the Gaussian smoothing technique for estimating the curvature of a digital contour. Differential operators of order n are obtained as successive convolutions of m (say) binomial kernels and n difference kernels. The authors have shown that this differential estimator is multigrid convergent for well-chosen m (m depends on the sampling rate and other parameters like maximal curvature), even in the presence of noise. The main problem with this method is how to choose this m for a given input shape, since there is an *ad hoc* balance to be made between accuracy and smoothness, and some parameters are tricky to estimate. This method is also computationnaly costly for large m . We will nevertheless use it to show that our reconstruction method is relatively independent of the curvature field estimation.

Visual Curvature (VC). The visual curvature has been introduced by Liu, Latecki and Liu [13]. Its principle is to measure the number of extreme points of the height function along several directions, within a given window around each point. It is thus clear that points on the shape boundary that are also vertex of its convex hull are always extreme points. Furthermore, they introduced a scale parameter which keeps only extreme points surrounded by big enough concave or convex parts. This process filters non-significant features at a given scale. Keeping only the vertices with a non-zero multiscale visual curvature defines a simplified polygon. The sequence of polygons obtained by increasing the scale parameter from 0 to 1 creates a natural filtration of polygons, the simplest one

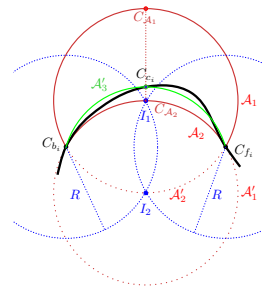
being the convex hull. A drawback of the method is that the visual curvature is only a qualitative estimation of the curvature in the general case. Furthermore, the simplified polygons are not controlled by an error measure. Lastly, this method requires four parameters. The visual curvature technique is recognized as a good feature detector and multiscale contour polygonalizer. Since the estimator is not suitable for our circle arc reconstruction (no distinction between concave/convex area), we will use it only for comparison purpose to assess the quality of our reconstruction method.

3 Contour Reconstruction with Circle Arcs

We propose a simple strategy to reconstruct a digital contour with the circle arc primitive. The main idea is to decompose the curvature estimation profile so as to find the significant curved areas which can be approximated by a circle arc. For this purpose a *split/merge* strategy is proposed. Split/merge is governed by a given maximal Hausdorff error E_{max} with respect to the input digital contour. This parameter acts as a scale for the reconstruction. It also induces specific parametrizations of curvature estimators, which will be described in the multi-scale reconstruction paragraph.

Algorithm 1 gives an overview of the main reconstruction process. First the curvature profile is decomposed into constant curvature parts. Then the set of local maxima/minima is extracted to define the initial regions as circle arcs. Since the circle arc estimation does not guarantee an error smaller than E_{max} , a split process is first proposed to reduce the error between the circle arc and the curve until it becomes less than the maximal allowed value. The merge phase extends them with their neighborhood regions while the associated circle arc gives an error lower than E_{max} . Before describing the error measure we focus on the problem of arc reconstruction from a contour region.

Arc reconstruction. Given a contour region R_i there are several ways to reconstruct a circle arc \mathcal{A}_i from two endpoints C_{b_i} and C_{f_i} . A first solution is to use the curvature information to determine the different possible solutions if there exist. Such a reconstruction is illustrated on the following figure by assuming a constant curvature value $\kappa_i = \frac{1}{R}$ estimated on the contour between C_{b_i} and C_{f_i} . The two possible centers of the osculating circles of radius R are represented with the intersections I_1 and I_2 of the two dotted circles of center C_{b_i} and C_{f_i} . From these two points I_1 and I_2 , four circle arcs $\mathcal{A}_1, \mathcal{A}'_1$ and $\mathcal{A}_2, \mathcal{A}'_2$ are deduced as potential candidates for the reconstruction. Then the sign of the curvature retains only the circle arcs of same convexity. To select the final solution, we compute the distance between each middle arc point with the middle contour point C_{c_i} . In the previous illustrating example, the final arc will be \mathcal{A}_2 since the euclidean distance between C_{c_i} and $C_{\mathcal{A}_2}$ is less than the one between C_{c_i} and $C_{\mathcal{A}_1}$.



A second method of reconstruction simply assumes that the circle arc should interpolate the middle point C_{c_i} of the contour region. In this case the center of the circle arc can be determined if the center point is not collinear with the two other points C_{b_i} and C_{f_i} . By referring to the previous example the reconstructed circle arc is given by \mathcal{A}'_3 . The two approaches have been experimented and the latter one gives the best results.

Error measure. Through the reconstruction process we need to evaluate the precision of the approximation by computing the error made between the reconstructed circle arcs (\mathcal{A}_i) of extremities (C_{b_i}, C_{f_i}) and the pieces of digital contour (\mathcal{C}_i) defined between the two points. We propose to use the Hausdorff distance $\delta_H(\mathcal{A}_i, \mathcal{C}_i)$ defined as:

$$\delta_H(\mathcal{A}_i, \mathcal{C}_i) = \max\{\max_{b \in \mathcal{C}_i} \{\min_{a \in \mathcal{A}_i} d(a, b)\}, \max_{a \in \mathcal{A}_i} \{\min_{b \in \mathcal{C}_i} d(a, b)\}\}$$

The first term $\max_{b \in \mathcal{C}_i} \{\min_{a \in \mathcal{A}_i} d(a, b)\}$ is computed in linear time by taking into account for each point C_j the angle $\theta_{\mathcal{A}_i}$ of the circle arc and the angle θ_{C_j} of the contour point C_j with the segment $O_i C_{f_i}$ where O_i is center of the circle arc \mathcal{A}_i . Two cases need to be considered (see Fig. 1(a)). If $\theta_{C_j} < \theta_{\mathcal{A}_i}$ then the minimal distance to the arc can be defined as the distance between C_j and the projection C'_j of C_j on the this circle. The minimal distance between C_j to any point of \mathcal{A}_i is thus given by $R - \|O_i C_j\|$ where R is the radius of the circle of center O_i . If θ_{C_j} is greater than the circle arc angle $\theta_{\mathcal{A}_i}$ then the projection C'_j is not located on the circle arc and the error is then defined by the minimal distance between $\|C_j C_{f_i}\|$ and $\|C_j C_{b_i}\|$.

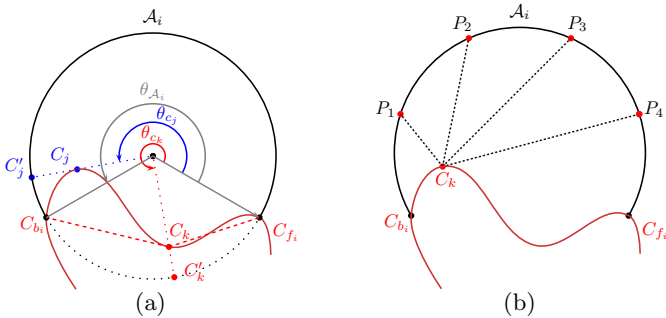


Fig. 1. Illustration of the evaluation of the Hausdorff distance between the circle arc and the digital contour

The second term of the Hausdorff error measure implies a $O(N^2)$ complexity since for any point of the arc, the minimum distance with the N points of the contour part $C_{b_i} C_{f_i}$ is required. We therefore only approximate this measure, by computing the minimal distance only for some sampling points on the circle arc. They are defined so as to obtain at least one sampling point on each quadrant (worst case for a circle arc with angle close to 2π). Thus four points are used to define the error (illustrated on Fig. 1(b)).

Merging process. The first merging process is initiated from the constant plot areas which are a local minima/maxima. The advantage of such strategy is to be independent of the choice of the initial contour point. Note that a precision parameter ϵ can be used to consider that two curvature values are equal. Then in order to avoid the presence of inconsistent circle arc, merging is allowed only between two contour regions with the same sign of curvature (definition included in the function `isExtendable`). It is also necessary to have a strategy for selecting in which order neighboring regions are merged with the considered central region. Here, the region whose mean curvature is closest is selected as first candidate to be merged. Such strategy is associated to the functions `selectFirstNearest`, `selectSecondNearest` and `extendFrontFirst` from Algorithm 3.

As described in Algorithm 4 the merging process is applied in a second phase in order to merge the potential arc regions located between two local maxima/minima. Finally, after this process, the function `selectMinErrPrimitive` is called to optionally verify if the straight segment could improve the reconstruction error. In order to favor circle arc reconstruction the straight segment is chosen only if it implies an error decrease at least equal to $E_{\max}/2$.

Algorithm 1. Reconstruction with arcs and segments

Data: $C = \{C_i\}_{i=0}^n$ digital curve, $\kappa = \{\kappa_i\}_{i=0}^n$ curvature estimation, float `maxArcError`;

Result: curve represented by a set of arcs and segments.

begin

Decompose κ into a set of constant curvature interval S defined by:

$\{(b_0, f_0), \dots, (b_i, f_i), \dots, (b_M, f_M)\}$.

For each contour point C_i , store in `regionIndex[i]` the index $k \in \{0, ..M\}$ of its region $S[k]$.

Extract from S the set S_m containing all the regions which are a local maxima/minima.

$S_{tmp} = S$;

while `nbElements(S_{tmp})` $\neq 0$ **do**

└ $S_{tmp} = \text{SPLIT_REGIONS}(S_{tmp}, S, \text{regionIndex}, \text{maxArcError})$;

// First extension from mini/maxima regions:

while `nbElements(S_m)` $\neq 0$ **do**

└ $S_m = \text{EXTEND_PLOT_REGIONS}(S_m, S, \text{regionIndex}, \kappa, \text{maxArcError})$;

// Second extension from all others regions S_u :

$S_u =$ set of index of valid non maxima/minima regions of the current regions.

while `nbElements(S_u)` $\neq 0$ **do**

└ $S_u = \text{EXTEND_PLOT_REGIONS}(S_u, S, \text{regionIndex}, \kappa, \text{maxArcError})$;

// Verify or change primitive for region which are better approximate

// by a straight segment:

└ `checkBestPrimitive($S, \text{tabRegionIndex}, \text{maxArcError}$)`;

end

Algorithm 2. SPLIT_REGIONS

Data: The set `setToCheck` of region index to be checked for splitting.
The set `S` of all curve `PlotRegions`.
The `rIndex` associating each point C_i to its region $S[k]$.
`maxArcError`: the maximal error allowed for splitting.
Result: A set containing all new index of splitted region.

```

for  $i = 0; i < \text{sizeOf}(\text{setToCheck}); i++$  do
    PlotRegion plotReg =  $S[i]$ ;
    float error = ComputeError(plotReg);
    if  $\text{error} > \text{maxArcError}$  then
        // Split the considered region
        PlotRegion newRegion1, newRegion2;
        newRegion1.b = plotReg.b;
        newRegion1.f =  $(\text{plotReg.b} + \text{size}(\text{plotReg})/2) \bmod (\text{rIndex.size}());$ 
        newRegion2.b =  $(\text{plotReg.b} + \text{size}(\text{plotReg})/2 + 1) \bmod (\text{rIndex.size}());$ 
        newRegion2.f = plotReg.f;
        add(S, newRegion1); add(S, newRegion2);
        add(splitSet, newRegion1); add(splitSet, newRegion2);
        InvalidAndUpdateReg( $S[i]$ , rIndex); updateRegs(newRegion1,
        newRegion2, rIndex);
return splitSet;

```

Multi-scale reconstruction. The reconstruction process is governed by a maximal error, which naturally induces a multi-scale reconstruction. To speed up the process and enhance the detection of significant points, the curvature estimator should also be tuned to take into account this error. This is easily done for the GMC curvature estimator, whose thickness parameter ν corresponds nicely with the Hausdorff error E_{\max} . Fig. 2(a) illustrates the blurred segment of width ν used in the GMC curvature estimator. When using the GMC estimator, the maximal error and the thickness ν are simply set to the chosen scale value. For the reconstruction with the BCC estimator its mask size has a linear dependence with the chosen scale value (although the initial mask size must be somehow determined by the user it was set as the contour size for the following experiment), while the maximal error is set equal to the chosen scale.

Time complexity. The global reconstruction complexity is first dependant of the curvature estimator. Since the error measure is computed in linear time, the complexity for the split/merging process is in the worst case equal to $O(n^2)$ when the contour (composed of n points) is reconstructed with a final arc obtained by adding one point at each step. On average the complexity is equal to $O(n \log(n))$.

Beside the difference in quality of the reconstructions, which is described in the next section, the choice of GMC or BCC estimator influences the efficiency of the merging process. Fig. 2(b) shows the number of region merges obtained for both estimators. The resulting plot shows that GMC is around 7 times more efficient than BCC for the merging process. The main reason of this difference comes from the stability of the GMC estimator (see [11] for detailed comparisons).

Algorithm 3. EXTEND_REGIONS

```

Data: The set setToExtend of region index to be checked for merging.
The set S of all PlotRegions.
The rIndex associating each point  $C_i$  to its region  $S[k]$ .
The curvature estimation  $\kappa = \{\kappa_i\}_{i=0}^n$ ,
maxArcError: the maximal error allowed after a merge.
Result: The set of region indexes which were updated: resultSet
for  $i = 0; i < nbElements(setToExtend); i++$  do
    PlotRegion reg = SetToExtend[ $i$ ];
    PlotRegion regBack = getBackRegion(reg, S, rIndex);
    PlotRegion regFront = getFrontRegion(reg, S, rIndex);
    PlotRegion regFirst = selectFirstNearest(regBack, reg, regFront,  $\kappa$ );
    PlotRegion regSec = selectSecondNearest(regBack, reg, regFront,  $\kappa$ );
    bool frontFirst = extendFrontFirst(regBack, reg, regFront,  $\kappa$ );
    if is_Extendable(reg, regFirst,  $\kappa$ ) then
        PlotRegion tmp = MERGE(reg, regFirst, frontFirst);
        float error = COMPUTE_ERROR(tmp);
        if error < maxArcError then
            add(resultSet, tmp); add(S, tmp);
            InvalidAndUpateReg(reg, rIndex);
            InvalidAndUpateReg(regFirst, rIndex);
    if is_Extendable(reg, regSec,  $\kappa$ ) then
        PlotRegion tmp = MERGE(reg, regSec, !frontFirst);
        float error = COMPUTE_ERROR(tmp);
        if error < maxError then
            add(resultSet, tmp); add(S, tmp);
            InvalidAndUpateReg(reg, rIndex);
            InvalidAndUpateReg(regSec, rIndex);
return resultSet;

```

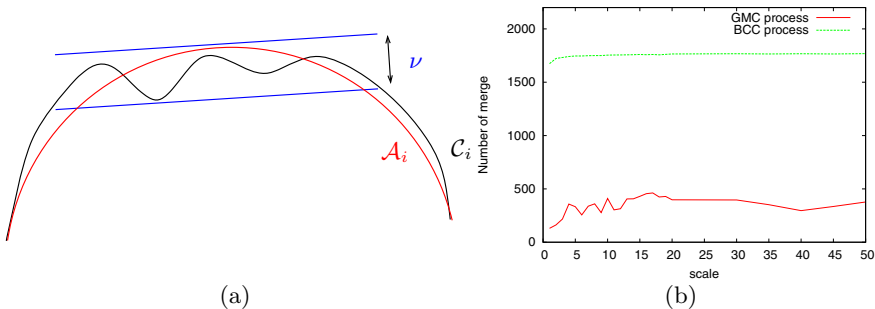


Fig. 2. Illustration of the blurred segment recognition of width ν illustrated in blue (a). A resulting arc detection at the scale $E_{max} = \nu$ is illustrated in red. Comparison of the number of merges of circle arcs when the reconstruction is defined with GMC estimator or with BCC estimator by using different scales (horizontal axis) (b).

4 Experiments and Comparisons

In this section, the experiments and comparisons were performed on a MacBook Pro running *Mac OS X 10.6.4*, with a processor 2.8 GHz *Intel Core 2 Duo* and 4GB of memory. Note that most of the experiments presented here are available online [12]: the reader may here tests and compares the different methods with its own shapes. First, we reconstruct the kangaroo shape of Fig. 4. Images (a-d) show the reconstructions obtained with the GMC estimator. For all experiments of the figure, resulting arcs are represented alternatively in blue and green color while straight segments are represented in red. The comparison between green GMC and BCC (images (e-h)) curvature estimators shows a relative equivalence of the reconstructions: approximately the same accuracy with the same number of primitives (slight advantage of accuracy for GMC). However BCC is much slower, especially at large scale. Note that the initial constant curvature intervals were defined from the same precision parameter ϵ set to $10e - 6$ for both GMC and BCC estimator.

We also compare our proposed method with the NASR method of Nguyen, who proposed a linear time algorithm for approximate circle arc recognition (chapter 4, page 133) [15]. The method is based on the representation of the contour in the tangent space and shares with the GMC estimator the preprocess with blurred segments (thickness is used as a scale parameter for comparisons). NASR method is faster than GMC and BCC approach (Fig. 4, images (i-l)), but is much less accurate at a comparable scale (i.e. for the same number of primitives). The Fig. 3 displays the number of primitives (segments and arcs) according to the scale. GMC and BCC show comparable evolution while circle arc primitives disappear in NASR at large scales.

As mentioned in Section 2, we also perform comparisons with the classical visual curvature reconstruction ([13], see Fig. 5), later called VC. Depending on its parameters, VC is generally faster than GMC, especially at large scale. However, for a given number of primitives, the GMC reconstruction always achieves better accuracy than VC (Hausdorff distance about 6-7 times smaller). Note that arcs and segments represent the same cost since each reconstructed arc interpolates the middle region point C_{C_i} (second method of arc reconstruction described in Section 3). To conclude the experimentation, a last comparison was performed on

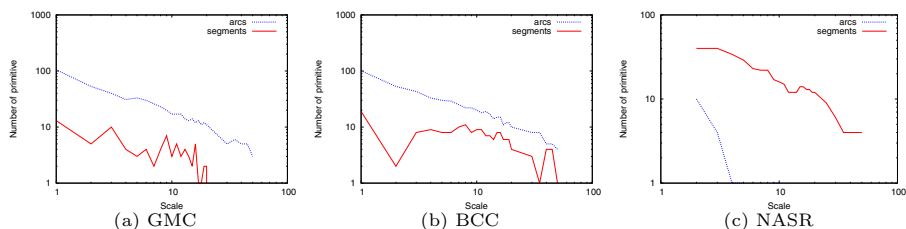


Fig. 3. Primitive evolution through the change of scale of the shape of Fig. 4

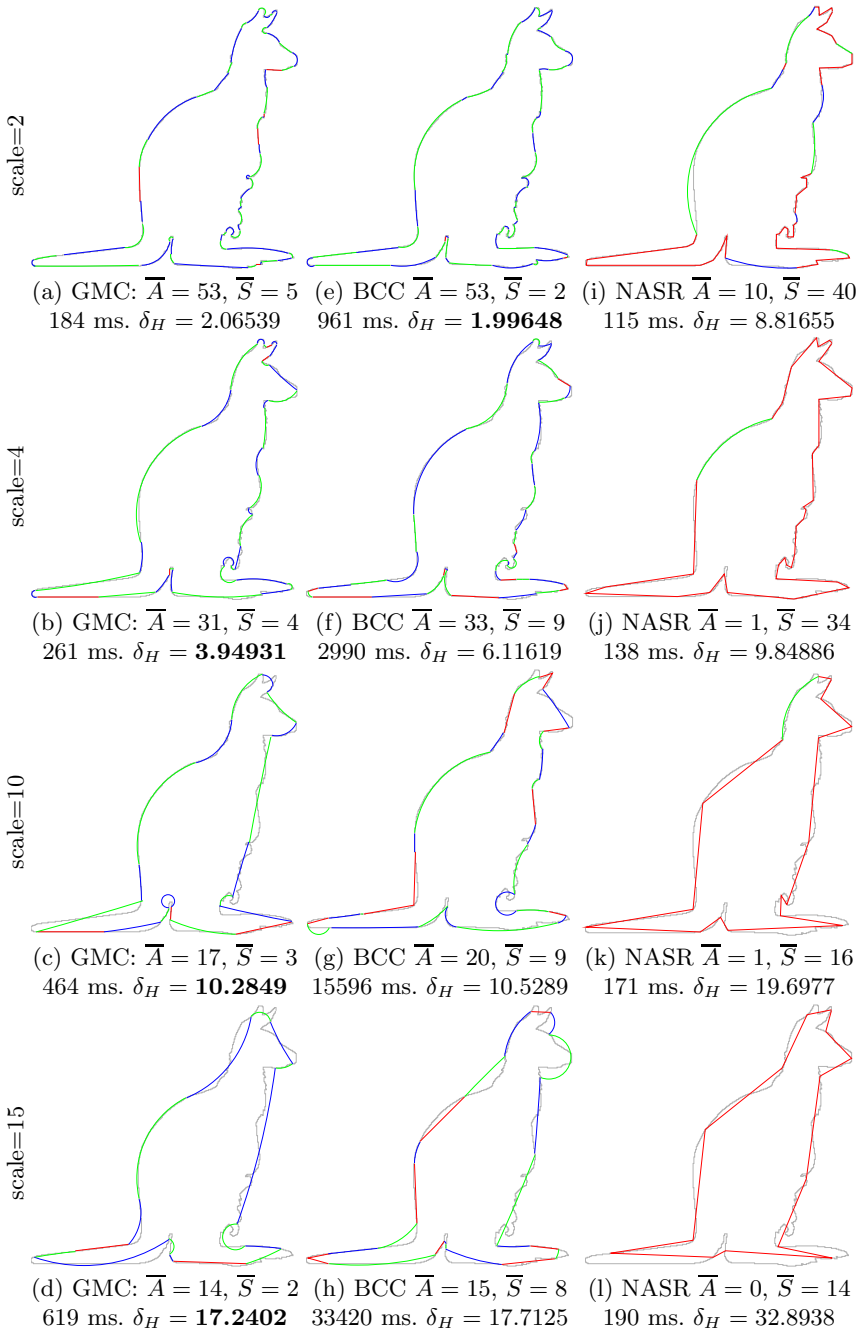


Fig. 4. Results and comparisons using the Hausdorff distance δ_H of the same reconstruction process by using GMC and BCC curvature estimators (a-h). Images (i-l) show for comparison, the result with NASR method by using the same scale parameter.

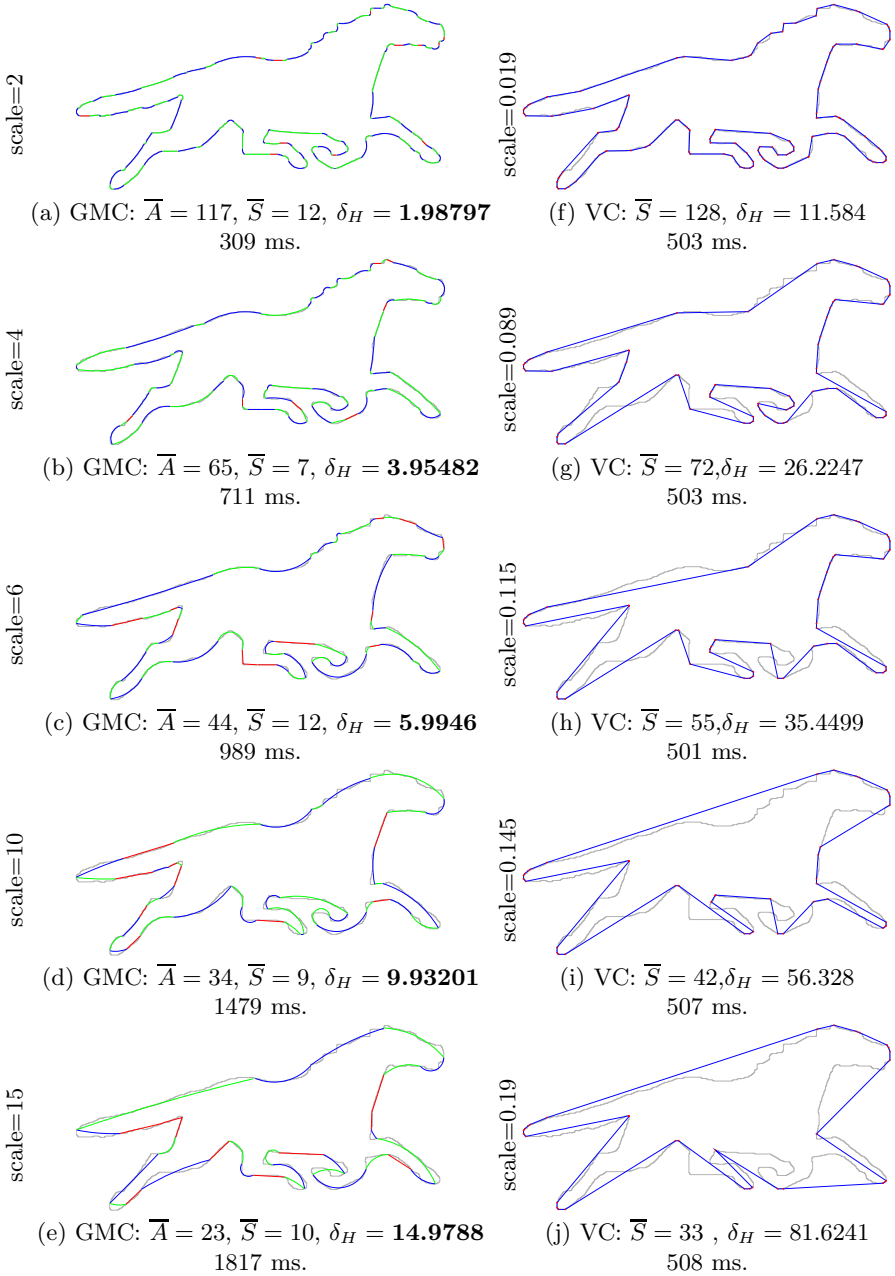


Fig. 5. Reconstruction results of the proposed method (a-e) at different scales. Comparisons with the Visual Curvature based approach.

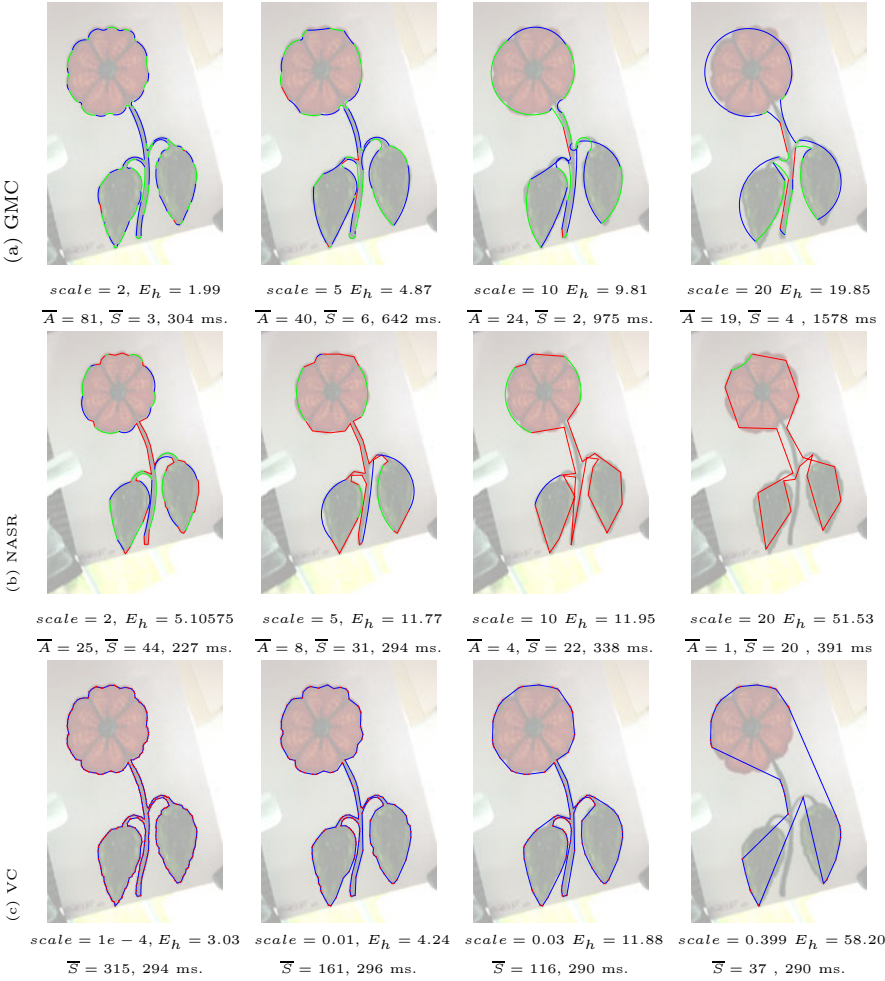


Fig. 6. Experiments and comparisons of three methods applied on a photography of a flower drawing. For GMC and NASR, the resulting circle arcs are represented alternatively with blue and green color while straight segments are represented in red.

a real photography of a drawing (Fig. 4) with the three different methods GMC, NASR and VC. The resulting representations confirm that GMC is always more precise for a given number of primitives.

5 Conclusion

This paper has proposed a simple method to reconstruct a digital contour with circular arcs, given a scale parameter that is simply the maximal Hausdorff error. Although the method is not specific to one curvature estimator, the GMC estimator has shown the best adequacy with our algorithm, since it gives precise results while keeping a reasonable execution time. The comparisons with the

recent works demonstrate the quality of the proposed reconstruction. In future works, we plan to integrate more information in the reconstruction process, namely the automatic noise detection and the information on flat/curve contour parts [10], in order to obtain a parameter free contour reconstruction.

References

1. Bretin, E., Lachaud, J.O., Oudet, E.: Regularization of discrete contour by willmore energy (2010) (submitted)
2. Chen, J.-M., Ventura, J.A., Wu, C.: Segmentation of planar curves into circular and line segments. *Image Vision and Computing* 14, 71–83 (1996)
3. Debled-Rennesson, I., Feschet, F., Rouyer-Degli, J.: Optimal blurred segments decomposition of noisy shapes in linear time. *Computers & Graphics* 30(1) (2006)
4. Bodansky, E., Gribov, A.: Approximation of a polyline with a sequence of geometric primitives. In: Campilho, A., Kamel, M.S. (eds.) *ICIAI 2006*. LNCS, vol. 4142, pp. 468–478. Springer, Heidelberg (2006)
5. Esbelin, H.-A., Malgouyres, R.: Convergence of binomial-based derivative estimation for C^2 noisy discretized curves. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 57–66. Springer, Heidelberg (2009)
6. Feschet, F.: Multiscale analysis from 1d parametric geometric decomposition of shapes. In: *IEEE (ed.) Int. Conf. on Pattern Recognition*, pp. 2102–2105 (2010)
7. Horng, J.H.: An adaptive smoothing approach for fitting digital planar curves with line segments and circular arcs. *Pat. Rec. Letters* 24(1-3), 565–577 (2003)
8. Horng, J.H., Li, J.T.: A dynamic programming approach for fitting digital planar curves with line segments and circular arcs. *Pat. Rec. Letters* 22(2), 183–197 (2001)
9. Kerautret, B., Lachaud, J.O.: Curvature estimation along noisy digital contours by approximate global optimization. *Pattern Recognition* 42(10), 2265–2278 (2009)
10. Kerautret, B., Lachaud, J.O.: Multi-scale analysis of discrete contours for unsupervised noise detection. In: Wiederhold, P., Barneva, R.P. (eds.) *IWCIA 2009*. LNCS, vol. 5852, pp. 187–200. Springer, Heidelberg (2009)
11. Kerautret, B., Lachaud, J.O., Naegel, B.: Curvature based corner detector for discrete, noisy and multi-scale contours. *IJSM* 14(2), 127–145 (2008)
12. Kerautret, B., Lachaud, J.O., Nguyen, T.P.: Curvature based contour representation demo (2010), <http://kerrecherche.iutsd.uhp-nancy.fr/CBContours>
13. Liu, H., Latecki, L.J., Liu, W.: A unified curvature definition for regular, polygonal, and digital planar curves. *Int. J. Comput. Vision* 80(1), 104–124 (2008)
14. Malgouyres, R., Brunet, F., Fourey, S.: Binomial convolutions and derivatives estimation from noisy discretizations. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) *DGCI 2008*. LNCS, vol. 4992, pp. 370–379. Springer, Heidelberg (2008)
15. Nguyen, T.P.: Etude des courbes discrètes: applications en analyse d’images. Ph.D. thesis, Nancy University - LORIA (2010) (in french)
16. Rosin, P.L., West, G.A.W.: Segmentation of edges into lines and arcs. *Image Vision Comput.* 7(2), 109–114 (1989)
17. Salmon, J.P., Debled-Rennesson, I., Wendling, L.: A new method to detect arcs and segments from curvature profiles. In: *ICPR*, pp. 387–390. IEEE, Los Alamitos (2006)
18. Tortorella, F., Patraccone, R., Molinara, M.: A dynamic programming approach for segmenting digital planar curves into line segments and circular arcs. In: *ICPR* (2008)

Recursive Calculation of Relative Convex Hulls

Gisela Klette

Auckland University of Technology,
School of Computing & Mathematical Sciences,
Private Bag 92006, Auckland 1142, NZ

Abstract. The relative convex hull of a simple polygon A , contained in a second simple polygon B , is known to be the minimum perimeter polygon (MPP). Digital geometry studies a special case: A is the inner and B the outer polygon of a component in an image, and the MPP is called minimum length polygon (MLP). The MPP or MLP, or the relative convex hull, are uniquely defined. The paper recalls properties and algorithms related to the relative convex hull, and proposes a (recursive) algorithm for calculating the relative convex hull. The input may be simple polygons A and B in general, or inner and outer polygonal shapes in 2D digital imaging. The new algorithm is easy to understand, and is explained here for the general case. Let N be the number of vertices of A and B ; the worst case time complexity is $\mathcal{O}(N^2)$, but it runs for “typical” (as in image analysis) inputs in linear time.

Keywords: relative convex hull, minimum perimeter polygon, minimum length polygon, shortest path, path planning.

1 History and Outline of the Paper

Studies about relative convex hulls started in the 1970s in image analysis, robotics and computer vision. Sklansky [6] proposed the notion of the relative convex hull at a time when digital imaging was still in its infancy, and digital images of low resolution. The relative convex hull is known to be identical to the minimum perimeter polygon (MPP). This polygon circumscribes a given set within an available (polygonal) domain. Figure 1 shows on the left an inner polygon A , an outer polygon B , and the convex hull of A relatively to B ; the image on the right in this figure illustrates stacked cavities for those two polygons.

The relative convex hull generalizes the concept of the convex hull. Algorithms for the computation of convex hulls of simple polygons are basic procedures in computational geometry or digital image analysis. For example, Melkman [4] proposed a linear time algorithm for connected simple polylines; a *simple polyline* consists of subsequent (connected) line segments, without any crossing. The end vertex of one line segment coincides with the start vertex of the following line segment, possibly apart from the start and end vertex of the simple polyline (i.e., the simple polyline does not need to form a loop). A *simple polygon* is defined by a simple polyline if this forms a loop. The polyline is then the *frontier* of this polygon. The Melkman algorithm works efficiently for any simple polyline by

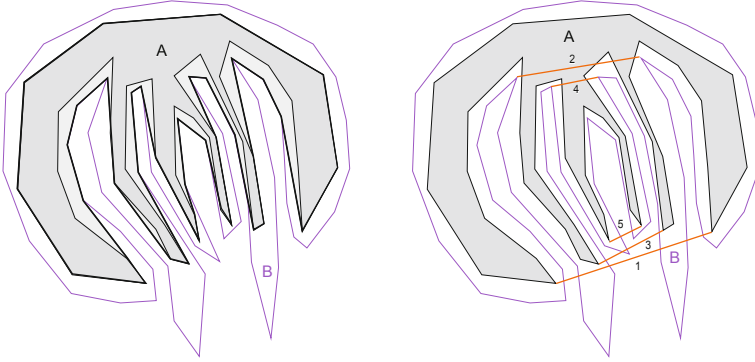


Fig. 1. An example of polygons A and B having stacked cavities. Left: Convex hull of A relatively to B (i.e., the *relative convex hull*) shown by the bold line. Right: Covers of stacked cavities.

using a *deque* (i.e., a double-ended queue) while computing the convex hull. We use this efficient algorithm as part of our new algorithm for the computation of the *relative convex hull* of a simple polygon. This convex hull is defined relatively to a second (larger) simple polygon.

Planning a shortest path for a robot in a restricted two-dimensional (2D) environment may also be solved by computing the relative convex hull of one simple inner polygon with respect to an outer simple polygon: The outer polygon might be defined by corners (vertices) of the available area, and the inner polygon by corners (vertices) of the obstacles. Obviously, inner and outer polygons in this case differ from polygons in image analysis. Here, vertices are in the real plane, edges are not limited to be isothetic, and the “width” of the space between inner and outer polygon is not constrained a-priori. Some publications about relative convex hulls are discussing geometric properties, but not algorithms, such as [8] or [9]. Proposed algorithms will be briefly reviewed below, before discussing a new algorithm. The paper is restricted to simple polygons. (The inner polygon could also be replaced by sets of points or other data.)

The new algorithm is for the general case of simple polygons (such as in the robotics scenario), but also for more constrained polygons such as in the digital imaging example. The relative convex hull provides a way of multigrid-convergent length measurement in digital imaging [2], thus making full use of today’s high resolution image data. In this particular context, the MPP is called *minimum length polygon* (MLP), and defines a multigrid-convergent method for length estimation. This method is an alternative way to purely local counts (e.g., counting edges along a digital frontier of an object). Local counts do not support more accurate length estimates in general when increasing the grid resolution. The advantage of having high resolution camera technology available would be wasted in image analysis if a method is applied that is not multigrid convergent. For showing multigrid convergence of the MLP towards the true perimeter, assume that a measurable set in the Euclidean plane is digitized

(Jordan digitization; see [2]) into an inner and an outer polygon. The relative convex hull (or MLP) is then calculated such that it contains the inner polygon, but itself is contained in the outer polygon. It has been shown [3] that the perimeter of this relative convex hull is converging towards the perimeter of the digitized measurable set with increasing the grid resolution.

First, the paper briefly recalls basic definitions and properties that are preliminaries to explain the algorithmic methods. We review existing algorithms for the computation of the relative convex hull. Next we list and show new theoretical results. They allow us to propose a completely new algorithmic approach for calculating the relative convex hull (or the MLP).

2 Basics

A simple polygon is bounded by a circular chain of co-planar line segments such that two segments only intersect at end (or start) vertices, and only if they are consecutive segments in the given circular chain. This circular chain is also called the *frontier* of the simple polygon.

A subset $S \subset R^2$ of points is convex iff S is equal to the intersection of all half planes containing S . The *convex hull* $CH(S)$ of a set of points S is the smallest (by area) convex polygon P that contains S .

2.1 The Relative Convex Hull

A finite set of n points in the plane can be associated with the set of n vertices of a simple polygon. The convex hull of a simple polygon A is a simple polygon $CH(A)$, and this has a reduced number of vertices if A is not convex. A simple polygon is also called a *Jordan polygon* because its frontier is a Jordan curve [4].

Definition 1. *A cavity of a polygon A is the topological closure of any connected component of $CH(A) \setminus A$.*

For example, in Fig. 1, the only cavity of A (defined by cover number 1), contains three non-connected subsets of cavities of B . A simple polygon A with n vertices has at most $\lfloor n/2 \rfloor$ cavities; in the maximum case all the cavities are triangles. In general, cavities are notated by $CAV_i(A)$, where $i = 1, 2, \dots$ follows the order of vertices on the frontier of A . A polygon is non-convex iff it has at least one cavity. A cavity is again a simple polygon with a minimum of three vertices, bounded by three straight line segments or more. This polygon may be convex or non-convex.

Definition 2. *A cover is a straight line segment in the frontier of $CH(A)$ that is not part of the frontier of A .*

A cover separates a uniquely assigned cavity from the exterior of $CH(A)$. A cover is defined by two vertices p_s and p_e , the start vertex of a cavity and the end

¹ A Jordan curve separates two connected regions the plane, here the interior of the polygon from the exterior of the polygon.

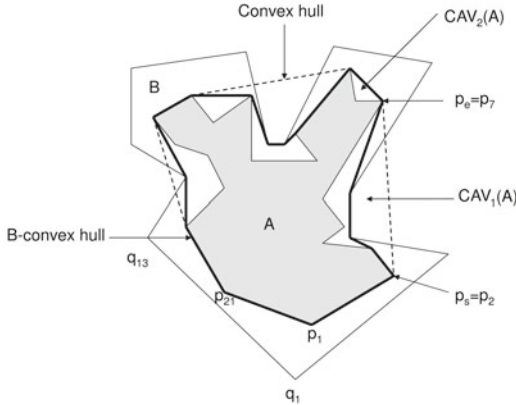


Fig. 2. The inner polygon A has three cavities with nonempty intersections with cavities of the outer polygon B . Cavity $CAV_2(A)$ has an empty intersection with cavities in B .

vertex of a cavity. The maximum depth of stacked cavities for the polygons A and B shown in Fig. 2 is two.

Let A be a simple polygon with n vertices, $A = \langle p_1, p_2, \dots, p_n \rangle$, and let B be a simple polygon with m vertices, $B = \langle q_1, q_2, \dots, q_m \rangle$, with $A \subseteq B \subset \mathbb{R}^2$.

Definition 3. A polygon A is B -convex iff any straight line segment in B that has both end points in A , is also contained in A . The convex hull of A relatively to B (in short, the B -convex hull of A ; formally $CH_B(A)$) is the intersection of all B -convex polygons containing A .²

The *minimum length polygon* (MLP) of a 2D digital object coincides with the relative convex hull of an inner grid polygon relatively to an outer grid polygon, normally defined in a way like simulating an inner and outer Jordan digitization. In 2D digital imaging, an object is a digitization of a measurable set $S \subset \mathbb{R}^2$ into a regular grid. We assume that a 2D picture P is composed of equally sized squares, where edges have length 1 and centers have integer coordinates. The inner polygon A is the union of all grid squares completely contained in the topological interior of a given object $S \subseteq \mathbb{R}^2$. The outer polygon B is the union of all grid squares having a nonempty intersection with the set S and the inner polygon A . The unknown frontier of the set S is assumed to be a Jordan curve γ located between the frontiers of polygons A and B .

The relative convex hull of A relatively to B (i.e., the MLP) is also located between the frontiers of polygons A and B , and its length is an approximation of the length of γ that converges to the true length of γ with increasing grid resolution [2]. It is uniquely defined for a given digitized object. The inner and the outer polygon of a Jordan digitization have the constraint that they are at

² This definition can also be generalized to higher dimensions.

Hausdorff distance 1. Mappings exist between vertices and between cavities in A and in B . The MLP may be calculated in linear time [1], where “linear” refers (e.g.) to the total number of grid squares between the inner and outer polygon.

In the general case, a simple polygon A inside of a simple polygon B , does not necessarily satisfy those constraints or properties, introduced due to the specifics of the regular grid. For the general case it is known that for all simple polygons A and B , only convex vertices of polygon A and only concave vertices of polygon B are candidates for the vertices of the relative convex hull [8].

Recently [5], an *arithmetic MLP* (AMLP) and a *combinatorial MLP* (CMLP) were proposed with the purpose of designing new algorithms for the computation of MLP [3]. For briefly discussing the AMLP, we consider the Gauss digitization of sets $S \subset \mathbb{R}^2$: All grid squares with their centroids in S belong to the resulting digital object. A set of connected grid squares is also called a *polyomino*. To be precise, a digital object is a polyomino iff it is 4-connected and the complement is 4-connected. The frontier of such a digitized object is a Jordan curve consisting of grid edges that separate the interior of the object from the exterior; these isothetic edges can be encoded by a Freeman chain code. The arithmetic definition is based on the fact that the tangential cover of the frontier of a digital object is a sequence of maximal digital straight lines (MDSS). The frontier of a digital region can always be uniquely divided into MDSS, assuming the start point is fixed (e.g., uppermost, leftmost). The object is *digitally convex* iff each pair of adjacent MDSSs of the tangential cover takes a convex turn. There are four different types of connected MDSSs with a single point overlap, called *zones*. A convex (concave) zone is an inextensible sequence of MDSSs where each pair of consecutive MDSSs takes a convex (concave) turn. The so-called *inflexion zones* are those parts where a convex turn is followed by a concave turn, or a concave turn is followed by a convex turn.

For a connected part of the frontier with only two kinds of steps (note: a digital straight line includes only two kinds of steps, c and $c + 1 \pmod{4}$) the left envelope (resp. right envelope) is the sequence of straight lines of the convex hull of the associated vertices of the inner polygon (outer polygon). It is a polygonal line starting at v_i and ending at v_j clockwise (counterclockwise). The AMLP is a closed polygonal line defined by those zones that may be convex, non-convex, or a segment connecting a vertex of the outer polygon with a vertex of the inner polygon, or a segment connecting a vertex of the inner polygon with a vertex of the outer polygon. The CMLP is based on a combinatorial approach (for details see [5]). Both definitions are equivalent to the MLP, however they lead to different linear time algorithms.

³ The authors of [5] motivate their work by a claim that the algorithm in [1] is faulty. Actually, it appears that the theoretical preparation of the algorithm in [1] in Euclidean space (when using Euclidean straight segments between vertices of the inner and outer polygon) is correct, but when using digital straight segments (DSSs) for an efficient implementation of the algorithm in the grid, only a simplified version of a DSS algorithm (DSSs up to complexity level 2 only) is used.

2.2 Algorithms for Calculating Relative Convex Hulls

Computing the relative convex hull of one simple polygon with respect to another simple polygon can be done in nearly (see below) linear time. In [11], the author changes the task to a shortest path problem between two vertices in a simple polygon by considering one extreme vertex of the inner polygon as start and end vertex at the same time, and by cutting $B \setminus A$, thus introducing a new (double-oriented) edge and combining outer and inner polygon into a single polygon. Then he proposes the triangulation of this resulting polygon $B \setminus A$, followed by finding the shortest path. The triangulation can be done in $\mathcal{O}(n \log \log n)$ time [10], and finding the shortest path can be done in linear time, using this triangulation. However the triangulation is a very complex operation, and [11] provides no further details besides such a rough sketch.

The algorithm in [1] computes the MLP in linear time with respect to the total number of vertices. It traces the inner polygon (or outer polygon) counter-clockwise, saves the coordinates of all convex or concave vertices in a list, and it marks each vertex with a plus sign if it takes a positive turn (convex), and a minus sign if it takes a negative turn (concave). Collinear vertices are ignored because they are not candidates for the MLP. All concave vertices of A are replaced by the concave vertices of B , by applying the (existing!) bijective map between those vertices. This step can be done in linear time. In a second run, the algorithm starts at a known MLP-vertex and it computes the next vertex of the relative convex hull by checking its location between the negative sides (straight line between known MLP-vertex and next concave vertex) and the positive sides (straight line between known MLP-vertex and next convex vertex). This step needs only linear time because candidate vertices need only be checked once. However the number of computations could be reduced by copying those vertices into the final list of the relative convex hull that belong to the convex hull of the inner polygon. (See the footnote on the previous page.)

Linear time algorithms for the computation of AMLP and CMLP are given in [5]. The authors point out that those algorithms are simpler than existing ones, and that they are easier to implement. The algorithm for the computation of the AMLP computes first the tangential cover of a digitized object in linear time. The tangential cover is decomposed into zones (convex, concave, inflexion), and for each zone the vertices of the convex hulls of the polygonal line of the zone are computed using the algorithm in [4], and then added to a list. Both algorithms work only for polyominoes, and the number of different zones increases with the number of cavities.

3 Theoretical Results

We state the following for highlighting two obvious facts:

Proposition 1. *The B -convex hull of a simple polygon A is equal to the convex hull of A iff the convex hull of A is completely contained in B (i.e., $\text{CH}(A) \subseteq B$), or if B is convex.*

On the other hand, the B-convex hull of a simple polygon A is different to $\text{CH}(A)$ if there exist at least one cavity in A and one cavity in B such that the intersection $\text{CAV}_i(A) \cap \text{CAV}_j(B)$ is not empty.

Theorem 1. *All vertices of the convex hull of a simple polygon A inside a simple polygon B are vertices of the B-convex hull of A .*

Proof. First we consider start and end vertices p_s and p_e of one fixed cavity $\text{CAV}(A)$. They are per definition vertices of the convex hull of A . If the intersection $\text{CAV}(A) \cap \text{CAV}(B)$ is empty then B has no vertices inside the cavity of A , and because $A \subseteq B$, the straight line between p_s and p_e connects vertices in A and B . Assuming that $\text{CAV}(A) \cap \text{CAV}(B)$ is not empty then there is at least one vertex $q_i \subseteq B$ inside the cavity of A , and the straight line between p_s and p_e crosses the exterior of B . A polygonal line between p_s and p_e , connecting the vertices of B inside the cavity, exists such that all of its line segments do not cross the frontier of A , and also do not cross the frontier of B .

Now we consider two consecutive vertices of $\text{CH}(A)$ that are not the start or the end of a cavity; they belong to the $\text{CH}_B(A)$ per definition. □

We assume that the convex hull of A is saved in a deque $D(A)$ after the application of the Melkman algorithm. We can copy those vertices to the B-convex hull of A , and we have to insert additional vertices into the deque $D(A)$ between p_s and p_e , for each cavity in A .

Let us consider cavities in polygons A and B with $\text{CAV}(A) \cap \text{CAV}(B) \neq \emptyset$. Let I_{new} be the polygon inside the cavity of A that is defined by vertices p_s and p_e in A and all the vertices in B located inside this cavity of A (see Fig. 3 with $I_{new} = \langle p_s, q_3, q_4, \dots, q_{11}, p_e \rangle$).

Theorem 2. *All the vertices of the convex hull of I_{new} belong to the relative convex hull of A between p_s and p_e .*

Proof. We assume that I_{new} has no cavity. Then I_{new} is convex and all the straight lines between vertices of I_{new} are inside the convex hull of A , and they do not cross the exterior of B . They do not cross A because $A \subseteq B$. If I_{new} has a cavity with vertices of B inside the cavity then the vertices p_s and p_e on I_{new} belong obviously to the convex hull of this polygon. The straight lines between those vertices are completely in $\text{CH}(A)$ and in B (see q_{10}, q_{11}, p_{10} in Fig. 3). If vertices of A are inside the cavity (see p_2, p_3, q_6 in Fig. 3) then the straight line between vertices p_s and p_e would cross A . A polygonal line starting at p_s and ending at p_e connects vertices in A . □

If the new polygon I_{new} has a cavity $\text{CAV}(I_{new})$ with vertices of A inside, then the start and the end vertex of $\text{CAV}(I_{new})$ and the vertices of O_{new} constitute again a new inner polygon, and the start and the end vertex of $\text{CAV}(I_{new})$ and the vertices of I_{new} constitute a new outer polygon (see Fig. 3). This establishes the basic idea of the new (recursive) algorithm.

We follow, in counterclockwise order, both frontiers of the original polygons. The computation of convex hulls starts usually (in computational geometry)

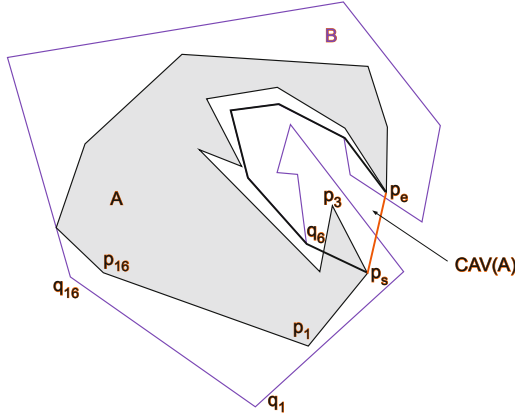


Fig. 3. Polygon A has one cavity. I_{new} has one cavity with one vertex of A inside.

at vertices with extreme values because it is obvious that vertices of A with extreme values (maximum and minimum y-coordinate, maximum and minimum x-coordinate) belong to the convex hull $CH(A)$; those extrema are also vertices of the relative convex hull $CH_B(A)$. Furthermore, for the computation of the convex hull and for finding overlapping cavities in A and in B , we use the fact that a vertex p_i of a polygon is convex if the frontier takes a positive turn, that means the determinant $t(p_{i-1}, p_i, p_{i+1}) > 0$, and analog a vertex is concave if the value of the determinant is negative. A vertex is collinear if the value of the determinant is zero.

In the digital image case, there exists a bijective mapping between cavities in A and in B for the inner and outer Jordan digitization polygons. This mapping is a useful constraint for the computation of the MLP. It simplifies the process of finding overlapping cavities in A and B in this special situation.

4 Recursive Algorithm

Our new algorithm is a recursive procedure. The base case of the recursion, where it stops, is the triangle. We use Theorem 1: The relative convex hull $CH_B(A)$ for simple polygons $A \subseteq B$ is only different from $CH(A)$ if there is at least one cavity in A and one in B such that the intersection of those cavities is not empty. The algorithm copies vertices of the convex hull of the inner polygon one by one until it finds a cavity. If it detects a cavity in A , then it finds the next cavity in B that has a nonempty intersection with the cavity in A , if there is any. The algorithm computes the convex hull of the new inner polygon I_{new} (all vertices of B inside the cavity of A , also the start vertex of $CAV(A)$, and also the end vertex of $CAV(A)$). For each cavity in this convex hull, the algorithm computes the convex hulls of the next new polygons. If there is no cavity

remaining, then it inserts the computed vertices inside the cavity, between p_s and p_e , for all cavities in A , and it returns the relative convex hull of the given polygon A relatively to B . - This is the basic outline, and we discuss it now more in detail.

We assume that vertices of A and B are given with their coordinates in a list. We compute the convex hulls of both polygons by applying the Melkman algorithm [4]. The computation for both polygons starts at the vertices with minimum y -coordinates. For a given ordered set of n vertices $A = \langle p_1, p_2, \dots, p_n \rangle$, the algorithm delivers the convex hull in a deque $D(A)$ where the first and the last element are the same vertices.

The difference between the indices of two consecutive vertices p_j and p_i in the resulting deque $D(A)$ is equal to 1 if there is no cavity between p_j and p_i . We do not change $D(A)$. A cavity in A with the starting vertex $p_s = p_j$ and the end vertex $p_e = p_i$ has been found if $(i - j) > 1$. The next step finds a cavity in the convex hull of the outer polygon. It searches the deque $D(B)$. If it finds a cavity, it checks if a straight line between the vertices of B crosses the straight line p_s and p_e , and it computes the convex hull for p_s, p_e and all the vertices of B that are left of $\overline{p_s p_e}$ and inside the cavity. If all vertices inside a cavity of B , including q_s and q_e , are on the right of straight line $\overline{p_s p_e}$, then this cavity of B does not intersect a cavity of A ; see $CAV_2(A)$ in Fig. 2. The relative convex hull does not change between p_s and p_e . We check the next cavity of B , with q_s on the right of $\overline{p_s p_e}$. The convex hull changes if q_s is on the right of $\overline{p_s p_e}$ and if one vertex between q_s and q_e , saved in the original list B , is on the left of $\overline{p_s p_e}$.

For example, consider Fig. 3. In this example the convex hull of A is saved in a deque and the set of vertices equals $D(A) = \langle p_1, p_2, p_{10} \dots p_{16}, p_1 \rangle$. We trace the deque. The difference between the first two indices is 1, we calculate the difference between the second and the third vertex. Between $p_2 = p_s$ and $p_{10} = p_e$ there must be a cavity because the difference of the indices is larger than 1. Vertices of B inside the cavity between p_s and p_e define now the new inner polygon with vertices $I_{new} = \langle p_s, q_3, q_4, \dots q_{11}, p_e \rangle$, and vertices of A inside the cavity between p_s and p_e define a new outer polygon with vertices $O_{new} = \langle p_s, p_3, p_4, \dots p_9, p_e \rangle$. All vertices of the convex hull $D(I_{new}) = \langle p_s, q_6, q_7, \dots q_{10}, p_e, p_s \rangle$ are vertices of the relative convex hull. The polygon I_{new} has one cavity p_2, p_3, q_6 with one vertex of polygon A inside. This defines again a new inner polygon. The convex hull of three vertices is always the same set of three vertices. The recursion stops. We replace p_2 and q_6 with p_2, p_3, q_6 in $D(I_{new})$. We continue to check for cavities until we reach p_e . The adjusted deque replaces the start and the end vertices in $D(A)$. In our example (see Fig. 3) the next cavity in $D(I_{new})$ starts at q_{10} and ends at p_e . But inside the cavity there is no element of the outer polygon. Thus, the relative convex hull does not change.

We continue to trace the deque $D(A)$ at p_{10} until p_1 is reached, and we skip vertex by vertex because there is no other cavity in A ; $D(A)$ stays unchanged. This concludes the example.

The following pseudo code provides the basic structure of the algorithm.

Algorithm 1. (Calculation of the relative convex hull)

Input: Simple polygons $A = \langle p_1, p_2, \dots, p_n \rangle$ and $B = \langle q_1, q_2, \dots, q_m \rangle$, $A \subseteq B$.

Output: Relative convex hull in $D(I)$ (counterclockwise).

- 1: Initialize $D(I) = \emptyset$, $D(O) = \emptyset$
- 2: Call Procedure $RCH(A, B, D(I), D(O), n, m, p_1, p_n)$

Note that $p_1 = p_{n+1}$ and $q_1 = q_{m+1}$. This algorithm applies recursively a procedure $RCH(I, O, D(I), D(O), l, t, p_s, p_e)$ that is sketched in Fig. 4.

- 1: Compute convex hulls of I and O in deques $D(I)$ and $D(O)$, respectively, l is number of vertices in I , t is number of vertices in O , p_s is the start vertex and p_e is the end vertex of the inner polygon. $k = 1$ and $j = 1$ are loop variables.
- 2: Remove the last elements in $D(I)$ and $D(O)$
- 3: **while** $k < l$ **do**
- 4: **if** Cavity between two consecutive vertices in $D(I)$, $(d_k$ and $d_{k+1})$ **then**
- 5: $p_s = d_k$ and $p_e = d_{k+1}$
- 6: **while** $j < t$ **do**
- 7: **if** Overlapping cavity between two consecutive vertices in $D(O)$ **then**
- 8: Update I such that I includes p_s and p_e and all vertices in O inside the cavity of I , L is the number of vertices
- 9: **if** $L > 3$ **then**
- 10: Update O such that O includes p_s and p_e and all vertices in I inside the cavity of I , T is the number of vertices
- 11: Call $RCH(I, O, D(I), D(O), L, T, p_s, p_e)$
- 12: **end if**
- 13: Insert q between p_s and p_e in $D(I)$
- 14: **end if**
- 15: **end while**
- 16: Return $D(I)$
- 17: **end if**
- 18: **end while**
- 19: Return $D(I)$

Fig. 4. Recursive procedure

The Melkman algorithm delivers the convex hull in a deque with the first element at the bottom of the deque and also at the top of the deque. We need to remove the elements from the top after the computation of convex hulls.

5 Discussion

Note that any vertex on A or B is accessed by the algorithm only at most as often as defined by the depth of a stacked cavity. This defines this algorithm as being of linear time complexity, measured in the total number of vertices on A and B , if (!) this depth is limited by a constant, but of quadratic time in the worst

case sense. For the general case, if there are “many” cavities in A , then they are all “small”, and the algorithm has again linear run-time behavior. If there is just one “big” cavity (similar to Fig. 3), then the recursion only proceeds for this one cavity, and we are basically back to the upper bound for the original input, but now for a reduced number of vertices. The worst case in time complexity is reached if there is a small number of “large cavities” in A and B , causing repeated recursive calls within originally large cavities of A , and the number of recursive calls defines the *depth* of those stacked cavities. However, such cases appear to be very unlikely in applications.

We discuss a few more details briefly for the example shown in Fig. 4. The cover $\overline{p_s p_e}$ of the first cavity cuts off three components of the original polygon B . The resulting simple polyline (from p_s to p_e) is not allowed to cross the segment $\overline{p_s p_e}$, see Fig. 5. left. We have two simple polylines from p_s to p_e , defining the inner and the outer polygon. Note that the inner polygon is now in general not simple, but this is not restricting the algorithm, because double edge orientations can only occur on $\overline{p_s p_e}$ (and the Melkman algorithm is for simple polylines anyway). For the second cavity, see Fig. 5. right. Here, the bold black line shows the calculated convex hull of the inner simple polyline, which defines again a new cavity.

Now consider the special case that A and B may be considered to be inner and outer digitizations of a set $S \subset \mathbb{R}^2$. The inner and outer polygons of a Jordan digitization satisfy special constraints. The step of finding the cavities with a nonempty intersection is faster because the algorithm can now use the existing [1] bijective map between cavities. Once we have found a cavity in A then there must be a cavity in B , and vertices for the new inner polygon are easy to find. First, the algorithm traces the inner polygon and the outer polygon counterclockwise and it saves the coordinates of all convex or concave vertices in a list. Then the recursive procedure returns the MLP. The maximum number of cavities in a polyomino with n convex or concave vertices is $\lfloor n/2 \rfloor$. In this maximum number case, the algorithm would stop after two loops.

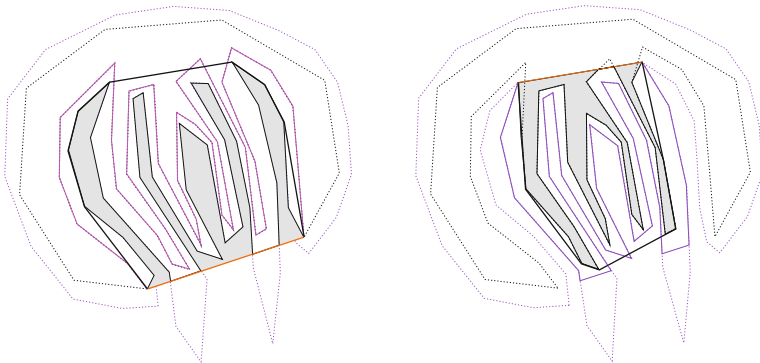


Fig. 5. First and second cavity for the example in Fig. 4

6 Conclusion

We presented a completely new algorithm for the computation of the B -convex hull of arbitrary simple polygons. It is a recursive procedure that is very simple and of low time complexity. The procedure uses a linear time algorithm for the computation of convex hulls, both for inner and outer polygons, starting with the given input polygons, and then continuing with the polygons defined in recursively defined cavities. The algorithm runs in linear time if the maximum depth of stacked cavities of A is limited by a constant. We continue to study the expected time complexity of the algorithm under some general assumptions of variations (i.e., distribution) for possible input polygons A and B .

References

1. Klette, R., Kovalevsky, V.V., Yip, B.: Length estimation of digital curves. *Vision Geometry*, SPIE 3811, 117–129 (1999)
2. Klette, R., Rosenfeld, A.: *Digital Geometry*. Morgan Kaufmann, San Francisco (2004)
3. Klette, R., Zunic, J.: Multigrid convergence of calculated features in image analysis. *J. Mathematical Imaging Vision* 13, 173–191 (2000)
4. Melkman, A.: On-line construction of the convex hull of a simple polygon. *Information Processing Letters* 25, 11–12 (1987)
5. Provençal, X., Lachaud, J.-O.: Two linear-time algorithms for computing the minimum length polygon of a digital contour. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 104–117. Springer, Heidelberg (2009)
6. Sklansky, J.: Measuring cavity on a rectangular mosaic. *IEEE Trans. Computing* 21, 1355–1364 (1972)
7. Sklansky, J., Kibler, D.F.: A theory of nonuniformly digitized binary pictures. *IEEE Trans. Systems, Man, and Cybernetics* 6, 637–647 (1976)
8. Sloboda, F., Stoer, J.: On piecewise linear approximation of planar Jordan curves. *J. Comput. Appl. Math.* 55, 369–383 (1994)
9. Sloboda, F., Zatko, B., Stoer, J.: On approximation of planar one-dimensional continua. In: Klette, R., Rosenfeld, A., Sloboda, F. (eds.) *Advances in Digital and Computational Geometry*, pp. 113–160 (1998)
10. Tarjan, R.E., Van Wyk, C.J.: An $\mathcal{O}(n \log \log n)$ algorithm for triangulating a simple polygon. *SIAM J. Computing* 17, 143–178 (1988)
11. Toussaint, G.T.: An optimal algorithm for computing the relative convex hull of a set of points in a polygon. In: *EURASIP, Signal processing III: Theories and Applications, Part 2*, pp. 853–856. North-Holland, Amsterdam (1986)

An Error Bounded Tangent Estimator for Digitized Elliptic Curves

Dilip K. Prasad, Raj Kumar Gupta, and Maylor K.H. Leung

School of Computer Engineering, Nanyang Technological University, Singapore
dilipprasad@gmail.com

Abstract. In this paper, we address the problem of tangent estimation for digital curves. We propose a simple, geometry based tangent estimation method for digital curves. The geometrical analysis of the method and the maximum error analysis for digital elliptic curves are presented. Numerical results have been tested for digital ellipses of various eccentricities (circle to very sharp ellipses) and the maximum error of the proposed method is bounded and is less than 5.5 degrees for reasonably large ellipses. The error for digital circles is also analyzed and compared with a recent tangent estimation method. In addition, the tangent estimation technique is applied to a flower shaped digital curve with six inflexion points and the results demonstrate good performance. The proposed tangent estimator is applied to a practical application which analyzes the error in a geometric ellipse detection method. The ellipse detection method is greatly benefited by the proposed tangent estimator, as the maximum error in geometrical ellipse detection is no more critically dependent upon the tangent estimation (due to the reduced error in tangent estimation). The proposed tangent estimator also increases the reliability and precision of the ellipse detection method.

Keywords: Tangent estimation, digital curves, error analysis, elliptic curves.

1 Introduction

Many image processing applications require tangent estimation for digital curves [1-10]. Examples include corner detection, inflection point detection, shape representation, Hough transform based methods for conic fitting, projective estimation of 3-dimensional shapes.

Geometric problems like estimating tangents, curvature, or shape features are well established for continuous non-digitized parametric curves. These problems become significantly difficult in the digitized pixel space of images, as the analytical equations may not take any continuous solution. The chosen solution is almost always an approximate integer solution nearest to the actual solution of the analytic equations. On one hand, digitization severely reduces the information contained in the continuous curves. On the other hand, the continuous curve can be analytically characterized using equation of finite number of coefficients (geometric parameters). Digitization introduces a non-linear corruption in the continuous curve which cannot be analyzed using equations [8-12].

Estimating the tangents in digitized curves is challenging because of three main reasons:

- The tangent is defined typically on a point, though it is a property of the continuous curve to which the point belongs. Thus, it has the local as well as the global properties of the curve. Due to the digitization, both these properties are affected and the nature or extent of effect cannot be quantified or analyzed using simple mathematical tools. At best, some estimates of maximum error or localized precision may be developed.
- Usually, while estimating the tangents, prior information about the nature of the curve is not available. Further, appropriate size of the local region around a point is also not known. Choosing these parameters is mainly heuristic based and not robust.
- Digitization permits many similar (though not the same) curves to be fit on the same digital pixel sequence. Thus, the tangents can never be obtained uniquely for the digital curves.

There are several approaches for tangent estimation in the case of digital curves. One of the methods to find the tangents is to use continuous function (typically second order) to approximate the curvature of the digital curve in a local region around the point of interest [13-14]. Then the derivative of the continuous function is used to determine the tangent. Such approach is restrictive in the choice of the nature of continuous function and the definition, shape, and dimension of the local region, etc. Besides being computationally intensive, they are also afflicted by the quantization noise. Further, there are applications where tangents need to be computed to fit a shape (for example ellipse) on the digital curve. In such cases, it is difficult to rely on a method that first fits a shape in the local region to estimate the tangent, and then uses the tangent to fit a shape to the whole curve. In order to overcome the problem of choosing the continuous function, researchers sometimes use a Gaussian filter to smoothen the digital curve and obtain a smooth continuous curve. This Gaussian smoothened continuous curve is then used for estimating the tangents [15].

Yet another method is to consider a family of continuous curves of various types. The whole digital curve is approximated by one of the continuous curves in the family using a global optimization technique. Then the tangents are computed on the curve chosen by optimization [16].

A different approach is to approximate the digital curves using line segments. Two main variations in this approach are in vogue. The first variation is based on the maximal segments [8, 11-12]. At the point of interest, the maximal line segments passing through it are found and weighted convex combination of their slopes is used to find the orientation of the tangent. Though this method is parameter-free, has asymptotic convergence, and incorporates convexity property, it is developed basically on heuristics, rather than analytic foundation.

Another variation is to approximate the digital curve using small line segments such that the maximum deviation of any point on the digital curve with one of the fitted line segments is small, below a threshold value of a few pixels [17]. This procedure divides the curve into small sub-curves each corresponding to a fitted line segment. Then the slope of the tangent at the midpoint of each sub-curve is

considered to be the same as the slope of the corresponding line segment. The main restriction with this method is that the tangents are available only at some points of the digital curves, viz., the mid points of the digital sub-curves.

We propose a tangent estimation method that is considerably simpler than all the above methods and has firm analytical foundation. Further, for estimating the slope of the tangent, only two points at a certain distance from the point of interest need to be found and no *a priori* information about the nature of curve is required. We prove that in the case of non-digitized ellipses, the slope computed by our method matches the slope of the actual tangent at any point on the curve. Though the proof is presented for elliptic curve only, it can be extended to all the conics. We considered elliptic curve because of its wide applicability. For the case of digital curve, we derive the expressions of the maximum error in tangent estimation. Based on the derivation, we compute the maximum error in the tangent estimation for a large range of ellipses, which include small circles to highly eccentric ellipses. The results sufficiently demonstrate the strength of the proposed tangent estimation method.

We also consider an example where this analysis of the upper bound of the error has significant influence. In the three point geometric Hough transform method [7], it was shown that the error in tangent estimation is the most important practical contributor of the error [1]. It was also shown that if the maximum error in computation is known, a reliability region for the computed centers can be predicted based upon a probability density function [1]. We use the maximum error for the proposed tangent estimation for this application.

In the following, Section 2 presents the proposed tangent estimation method and the geometrical proof of the concept. Section 3 analyzes the maximum error in the tangent estimation for digital curves using the proposed method. Section 4 presents numerical examples to illustrate the effectiveness of the proposed tangent estimator. Section 5 shows an application of the proposed tangent estimation method for ellipse detection. Section 6 concludes the article.

2 Proposed Tangent Estimation Method

In this section, we present the proposed tangent estimation method. The discussion has been restricted to elliptic curves. The analysis is easily extensible to any other conic as well. This section develops and tests the concept for the continuous curve only. We introduce the concept in section 2.1 and present the geometric proof of the validity of the concept in section 2.2.

2.1 The Concept

Let us consider an ellipse:

$$(x/a)^2 + (y/b)^2 = 1 \quad (1)$$

where, a and b ($a \geq b$) are the lengths of semi-major and semi-minor axes. Suppose we are interested in finding the tangent at the point $P_0(x_0, y_0)$. We know that the actual slope of the tangent at P_0 is given as follows:

$$m_0 = \frac{dy}{dx} = -\left(\frac{b}{a}\right)^2 \frac{x_0}{y_0} \tag{2}$$

In reality, since we do not know the curve to which P_0 belongs, we cannot compute the tangent analytically as above. We propose to use a small circle of radius $R \ll b$ centered at P_0 :

$$(x - x_0)^2 + (y - y_0)^2 = R^2 \tag{3}$$

The circle intersects the ellipse given by (1) at points P_1 and P_2 . There are two steps for finding the tangent at P_0 . First, find the slope of the line P_1P_2 (denoted by \tilde{m}). Second, find a line with slope \tilde{m} passing through the point P_0 .

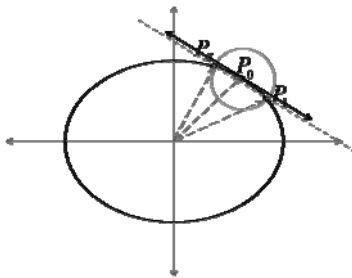


Fig. 1. Illustration of the geometric concept used for tangent estimation at the point P_0

The idea is demonstrated in Figure 1 above. The slope \tilde{m} of the line P_1P_2 is given by:

$$\tilde{m} = (y_2 - y_1) / (x_2 - x_1) \tag{4}$$

2.2 Geometric Proof of the Proposed Concept

For analysing the error in the proposed method we first need to find the coordinates of points P_1 and P_2 . For the ease of analysis, we define the coordinates of the points on the ellipse using a parametric notation as:

$$x = a \cos \theta; \quad y = b \sin \theta \tag{5}$$

Specifically, the coordinates of P_0 are given by $(a \cos \theta_0, b \sin \theta_0)$. It should be noted that θ is not a geometric angle. Then, we can substitute (5) and the coordinates of P_0 in (3) to find the intersection points P_1 and P_2 :

$$(a \cos \theta - a \cos \theta_0)^2 + (b \sin \theta - b \sin \theta_0)^2 = R^2 \tag{6}$$

The above equation can be simplified using algebraic and trigonometric manipulations as follows:

$$4 \sin^2 \left(\frac{\theta - \theta_0}{2} \right) \left(a^2 \sin^2 \left(\frac{\theta + \theta_0}{2} \right) + b^2 \cos^2 \left(\frac{\theta + \theta_0}{2} \right) \right) = R^2 \tag{7}$$

The solutions of the above equation gives the intersection points. The derivation for solving the above equation is very long and tedious. For brevity, we circumvent the derivation and state that for $R \ll b$, the value of θ is close to θ_0 :

$$\theta = \theta_0 + \Delta\theta; \quad \Delta\theta \approx 0 \tag{8}$$

Then (7) can be written as:

$$\begin{aligned} \lim_{\Delta\theta \rightarrow 0} \left\{ 4 \sin^2 \left(\frac{\theta - \theta_0}{2} \right) \left(a^2 \sin^2 \left(\frac{\theta + \theta_0}{2} \right) + b^2 \cos^2 \left(\frac{\theta + \theta_0}{2} \right) \right) \right\} \\ = 4 \sin^2 \left(\frac{\Delta\theta}{2} \right) \left(a^2 \sin^2 \theta_0 + b^2 \cos^2 \theta_0 \right) = R^2 \end{aligned} \tag{9}$$

Thus, $\Delta\theta$ can be computed as follows:

$$\Delta\theta = \pm 2 \sin^{-1} \left(\sqrt{\frac{R^2}{4(a^2 \sin^2 \theta_0 + b^2 \cos^2 \theta_0)}} \right) \tag{10}$$

Then the coordinates of P_1 correspond to the negative value of $\Delta\theta$, while the coordinates of P_2 correspond to the positive value of $\Delta\theta$, i.e. $\theta_2 + \theta_1 = 2\theta_0$, and the slope of the line P_1P_2 is given as follows:

$$\tilde{m} = \frac{b \sin \theta_2 - b \sin \theta_1}{a \cos \theta_2 - a \cos \theta_1} = -\frac{b}{a} \cot \left(\frac{\theta_2 + \theta_1}{2} \right) = -\frac{b}{a} \cot(\theta_0) \tag{11}$$

Substituting the parametric coordinates for x_0 and y_0 in (2), we get:

$$m_0 = \frac{dy}{dx} = -\frac{b}{a} \cot \theta_0 \tag{12}$$

We notice from (11) and (12) that in the analytical case (absence of digitization), if $R \ll b$, then there is no error in the computation of the slope with the proposed method.

2.3 Choice of R

The aim in choosing $R \ll b$ is that $\Delta\theta \approx 0$ and consequently $\sin(\Delta\theta/2) \approx 0$, such that (9) is valid. Suppose we choose a maximum value of $\Delta\theta$, denoted by $\Delta\theta_{\max}$, then using (10) and $\theta_0 = 0$, the expression for choosing R is given as:

$$R \leq 2b \sin(\Delta\theta_{\max}/2) \tag{13}$$

For example, if we use $\Delta\theta_{\max} = (\pi/18)$, i.e., 10° , such that $\sin(\Delta\theta/2) = 0.0872$, then $R \leq 0.1743b$.

3 Maximum Error in Tangent Estimation Due to Digitization

Due to digitization in the case of images, a general point $P(x, y)$ is approximated by a pixel $P'(x', y')$ as follows:

$$x' = \text{round}(x); \quad y' = \text{round}(y) \tag{14}$$

where $\text{round}(x)$ denotes the rounding the value of real number x to its nearest integer. $P'(x', y')$ satisfy the following:

$$x', y' \in \mathbb{Z} \tag{15}$$

$$x' = x + \Delta x; \quad y' = y + \Delta y \tag{16}$$

$$-0.5 \leq \Delta x \leq 0.5, \quad -0.5 \leq \Delta y \leq 0.5 \tag{17}$$

We shall use (16) and (17) for estimating the maximum error in the computation of the slope of the numeric tangent \tilde{m} . Let the slope of numeric tangent computed by pixels $P'_1(x'_1, y'_1)$ and $P'_2(x'_2, y'_2)$ (corresponding to P_1 and P_2) be denoted by \tilde{m}' . We shall call the numeric tangent computed with pixels as the digital tangent. Then \tilde{m}' can be solved as follows:

$$\tilde{m}' = \frac{y'_2 - y'_1}{x'_2 - x'_1} = \left(\tilde{m} + \frac{\Delta y_2 - \Delta y_1}{x_2 - x_1} \right) \Big/ \left(1 + \frac{\Delta x_2 - \Delta x_1}{x_2 - x_1} \right) \tag{18}$$

The angular difference between the numeric tangent and the digital tangent is used as the estimate of the error. This angular difference is given as:

$$\partial\phi = \tan^{-1}(\tilde{m}) - \tan^{-1}(\tilde{m}') = \tan^{-1}\left(\frac{\tilde{m} - \tilde{m}'}{1 + \tilde{m}\tilde{m}'}\right) \tag{19}$$

Substituting (18) in (19), we get:

$$\partial\phi = \tan^{-1}\left(\frac{\tilde{m}(\Delta x_2 - \Delta x_1) - (\Delta y_2 - \Delta y_1)}{(1 + \tilde{m}^2)(x_2 - x_1) + (\Delta x_2 - \Delta x_1) + \tilde{m}(\Delta y_2 - \Delta y_1)}\right) \tag{20}$$

Now based on the minimum and maximum possible values of $\Delta x'_1, \Delta x'_2, \Delta y'_1,$ and $\Delta y'_2$, we have the nine cases, corresponding to $(\Delta x_2 - \Delta x_1) \in \{-1, 0, 1\}$ and $(\Delta y_2 - \Delta y_1) \in \{-1, 0, 1\}$.

However, it can be shown that the maximum error occurs in cases $|\Delta x_2 - \Delta x_1| = |\Delta y_2 - \Delta y_1| = 1$:

$$\partial\tilde{\phi}_{\max} = \max\left(\frac{1}{s^3}(\sin\tilde{\phi} \pm \cos\tilde{\phi})\left(s^2 - s(\pm\cos\tilde{\phi} \pm \sin\tilde{\phi}) + (\pm\cos\tilde{\phi} \pm \sin\tilde{\phi})^2\right)\right) \tag{21}$$

where, $s = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ and $\tilde{\phi} = \tan^{-1}(\tilde{m})$. It should be noted that the above derivation of the error bound is applicable to any general continuous line

connecting any points P_1 and P_2 and the corresponding digital line with the slope \tilde{m}' . However, in the present case, P_1 and P_2 are given by (10). Thus, the error bound $\partial\tilde{\phi}_{\max}$ is related to R through s .

4 Numerical Results

In this section, we present numerical results for the maximum error due to the proposed tangent estimation method. Let us consider that the digital ellipses of interest have the length of semi-minor axis $b \geq 30$. Then, using section 2.3, a reasonable value of R is $R \leq 5.229$.

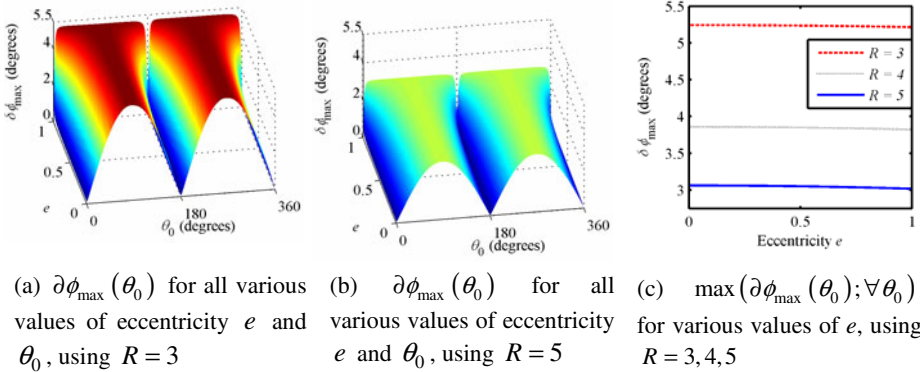


Fig. 2. Numerical results: maximum error in tangent estimation of digital ellipses using $b = 30$

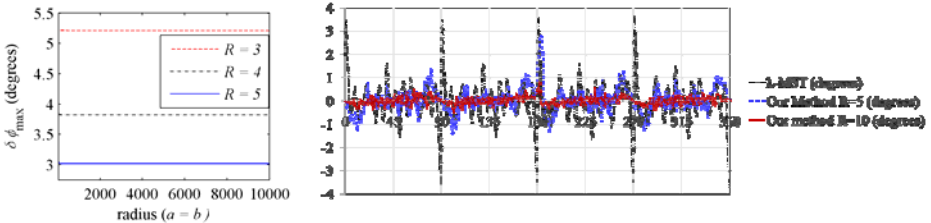
We perform three simulations, each with a different value of R , viz., $R = 3$ and $R = 5$ respectively. For each simulation we vary the eccentricity $e = \sqrt{(a^2 - b^2)}/a^2$ of the digital ellipse varies from 0 to 0.999 (circle to extremely sharp ellipses), while keeping $b = 30$, and compute $\partial\phi_{\max}(\theta_0)$. The result are presented in Figure 2 (a-b). In Figure 2(c), we present $\max(\partial\phi_{\max}(\theta_0); \forall \theta_0)$ for various values of eccentricities for $R \in \{3, 4, 5\}$. It is notable that the maximum error in the computation of the tangents is very small. It is less than 5.25° for $R = 3$ and less than 3.1° for $R = 5$.

Next, we consider digital circles whose radii range from 30 pixels to 10000 pixels. We use three different values of R , viz., $R = 3, R = 4$ and $R = 5$ respectively, and compute $\max(\partial\phi_{\max}(\theta_0); \forall \theta_0)$. The results shown in Figure 3(a).

In order to understand the difference between the observed maximum error for various values of R , let us consider the ratio of the maximum error due to digitization (which is equal to 0.5) to the radius R of the circle used for computing the tangent:

$(0.5/R)$. It is evident that this ratio is higher for $R = 3$ as compared to $R = 5$. Thus, the error in the computation of tangent using $R = 5$ is less than when $R = 3$ is used. If the minimum value of b is higher, such that we can choose larger value of R (satisfying (13)), the maximum error in the computation of tangents will further reduce.

For comparison with the recently proposed λ -MST estimator, we consider the example test proposed in [12]. Hundred experiments were performed, in each of which a digitized curve corresponding to radius 100 and a randomly chosen center within 1 pixel region were formed and absolute error in the computation of tangent was computed using λ -MST and the proposed method. The average error in the computation of tangent is shown in Figure 3(b). The results show that the proposed tangent estimation method has lower error than λ -MST estimator. Further, while the error in the λ -MST estimator is high at the shift of every quadrant, this feature is not strong in the proposed tangent estimation method.



(a) Maximum error in tangent estimation of digital circles with radius from $[30,10000]$ using different values of R

(b) Average absolute error in the computation of tangents for 100 experiments with digitized circles of radius 100 and centers within 1 pixel region chosen randomly. The result is compared with λ -MST estimator [12]

Fig. 3. Analysis of error for digitized circles

Finally, we show the error in tangent estimation for an analytical shape with inflexion points. We consider the flower shape with 6 petals:

$$x = \frac{400}{3}(1 - 1.5 \sin 6\theta) \cos \theta + 60; \quad y = \frac{400}{3}(1 - 1.5 \sin 6\theta) \sin \theta + 60 \quad (22)$$

The digitized shape is shown in Figure 4(a). The smallest circle enclosing this shape completely has a radius 200. The directions of the tangents on the actual and tangents computed on the digitized curve using $R = 20$ are shown in Figure 4(b). The plot shows a good agreement between the actual and computed tangents. The maximum and average errors for various values of R are plotted in Figure 4(c) We mention here that the maximum error occurs when the point of interest is close to the inflexion point. The error at the inflexion point itself is small (close to the average error) due to symmetry of the shape.

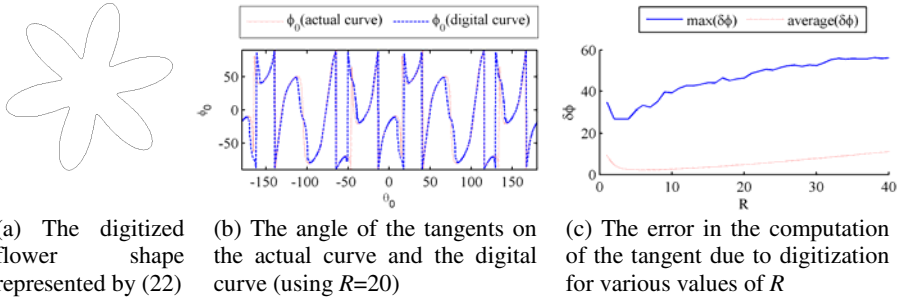


Fig. 4. Example of an analytical curve with inflexion points

5 Practical Application: Hough Transform Based Ellipse Detection Method (Yuen [7])

Yuen proposed a variation of randomized Hough transform for detecting ellipses that uses three points on a digital curve to first determine the center of the ellipse and then perform Hough transform [7]. This method was analyzed in [1] and it was shown that the error in the estimation of tangents is the most important contributing factor in this method. In the paper, the result were shown using $\partial\phi_{\max} = 15^\circ$, which was chosen empirically and is reasonable for the existing methods. However, for the proposed method, we see that the maximum error is less than 5.25° . Thus, the error contributed in Yuen’s method [7] for digital ellipses due to the tangent estimation is expected to decrease significantly.

In [1], among the four tests presented for analyzing the relative error r_{err}/a of ellipse fitting using Yuen’s method [7], Test 4(a,b) present the error contributed due to the erroneous tangent estimation. We substitute $\partial\phi_{\max} = 5.25^\circ$ and compute the results of Test 4(a,b) of [1] considering digitization. The results are presented in Figure 5. It can be seen that the error due to tangents is significantly less than the results presented in [1], which considered $\partial\phi_{\max} = 15^\circ$.

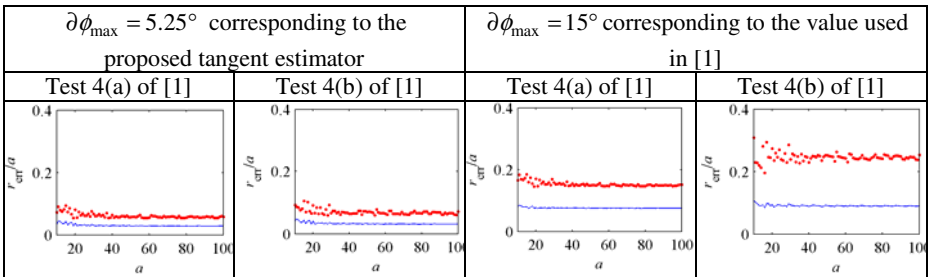


Fig. 5. Impact of the proposed method on the error analysis [1] for Yuen’s method [7]. The figures correspond to tests 4(a) and 4(b) of [1]. The dotted plot (red) shows the maximum error for the complete range of parameters, while the solid line (blue) shows the average error for the complete range for a particular value of $a \in [10,100]$.

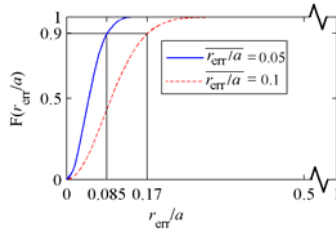


Fig. 6. The Rayleigh distribution function proposed in [1] for describing the probability density function of the relative error $\overline{r_{err}/a}$ in Yuen’s method [7]. The distribution for the proposed tangent estimator uses $\overline{r_{err}/a} = 0.05$, while $\overline{r_{err}/a} = 0.1$ was used in [1].

The value of $\overline{r_{err}/a} = 0.05$ obtained using the proposed tangent estimation method is very close to the $\overline{r_{err}/a}$ for the remaining tests in [1]. Thus, the error in estimation of tangents is no longer the most important contributing factor in the error analysis of [1]. In [1], the average value of relative error $\overline{r_{err}/a}$ is computed for four tests, and the maximum $\overline{r_{err}/a}$ among all the four tests is used to define probability density function (a Rayleigh’s function) as below:

$$F(r_{err}/a) = 1 - \exp\left(-\left(r_{err}/a\right)^2 / \left(2\sigma^2\right)\right) \tag{23}$$

where $\sigma = \overline{r_{err}/a} \sqrt{2/\pi}$. The Rayleigh distributions obtained using $\overline{r_{err}/a} = 0.05$ (corresponding to the proposed tangent estimation method, $\partial\phi_{max} = 5.25^\circ$) and $\overline{r_{err}/a} = 0.1$ (originally used in [1] considering $\partial\phi_{max} = 15^\circ$) are shown in Figure 6.

In the methods, where Yuen’s method [7] is used as a supplementary, or a part of the ellipse detection process, and the centers are computed using other methods (like least squares) as well, we can use (23) to choose a trust region. Suppose we want to verify the centers achieved using two methods, one of them is Yuen’s method, then we may say that the computed centers using the two methods are 90% reliable if:

- For $\overline{r_{err}/a} = 0.1$, the distance between the two centers is $r_{err}/a \leq 0.17$
- For $\overline{r_{err}/a} = 0.05$, the distance between the two centers is $r_{err}/a \leq 0.085$

Thus, the proposed tangent estimation method provides a stricter trust region and the centers can be computed more reliably as well as precisely using the proposed tangent estimation method.

6 Conclusion

A simple, geometry based method is proposed for estimating the tangents of digital curves. The proof of the geometric concept used in the method is also presented. In

addition, for digital ellipses, we perform the maximum error analysis and give explicit analytical terms for the maximum error. The maximum error in the tangent estimation using the proposed method is small (worst case 5.25° for the considered examples).

It is shown that the proposed tangent estimation method can have significant impact on some practical applications. For example, it is shown that if good tangent estimation methods like the proposed method are used in a specific application, the trust region for computing the centers of the ellipses (using various methods) can be chosen more strictly, which shall enhance the reliability as well as the precision of the ellipse detection method.

At present, the analysis and results have been presented for digital ellipses and circles only. An example of digital curve with inflexion points is also considered. It is easily extensible to other conics like parabola, hyperbole, etc. It can also be extended to quadric curves (fourth order curves). Work is in progress for comparing the performance of the proposed tangent estimation method with the other tangent estimation methods. In this comparison, we shall consider various performance criteria like precision, maximal error, isotropy, convergence, convexity on ideal digital shape, and time complexity [12]. Such study shall help the image processing community in choosing good tangent estimators suitable for their corresponding applications while understanding the strengths and limitations of the tangent estimator used by them.

References

1. Prasad, D.K., Leung, M.K.H.: Error analysis of geometric ellipse detection methods due to quantization. In: Fourth Pacific-Rim Symposium on Image and Video Technology (PSIVT 2010), Singapore (2010)
2. Anderson, I.M., Bezdek, J.C.: Curvature and tangential deflection of discrete arcs: a theory based on the commutator of scatter matrix pairs and its application to vertex detection in planar shape data. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI* 6, 27–40 (1984)
3. Coeurjolly, D., Klette, R.: A comparative evaluation of length estimators of digital curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26, 252–258 (2004)
4. Kovalevsky, V.: Curvature in digital 2D images. *International Journal of Pattern Recognition and Artificial Intelligence* 15, 1183–1200 (2001)
5. Lee, C.-K., Haralick, R.M., Deguchi, K.: Estimation of curvature from sampled noisy data. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, New York, USA, pp. 536–541 (1993)
6. Worring, M., Smeulders, A.W.M.: Digital curvature estimation. *Computer Vision and Image Understanding* 58, 366–382 (1993)
7. Yuen, H.K., Illingworth, J., Kittler, J.: Detecting partially occluded ellipses using the Hough transform. *Image and Vision Computing* 7, 31–37 (1989)
8. Feschet, F.: Canonical representations of discrete curves. *Pattern Analysis and Applications* 8, 84–94 (2005)
9. Faure, A., Buzer, L., Feschet, F.: Tangential cover for thick digital curves. *Pattern Recognition* 42, 2279–2287 (2009)
10. Tsai, D.M., Chen, M.F.: Curve fitting approach for tangent angle and curvature measurements. *Pattern Recognition* 27, 699–711 (1994)

11. De Vieilleville, F., Lachaud, J.O.: Experimental comparison of continuous and discrete tangent estimators along digital curves. In: Brimkov, V.E., Barneva, R.P., Hauptman, H.A. (eds.) IW CIA 2008. LNCS, vol. 4958, pp. 26–37. Springer, Heidelberg (2008)
12. De Vieilleville, F., Lachaud, J.O.: Comparison and improvement of tangent estimators on digital curves. *Pattern Recognition* 42, 1693–1707 (2009)
13. Lewiner, T., Craizer, M.: Projective estimators for point/tangent representations of planar curves. In: 21st Brazilian Symposium on Computer Graphics and Image Processing, SIBGRAPI 2008, Campo Grande, BRAZIL, pp. 223–229 (2008)
14. Lewiner, T., Gomes Jr, J.D., Lopes, H., Craizer, M.: Curvature and torsion estimators based on parametric curve fitting. *Computers and Graphics* 29, 641–655 (2005)
15. Mokhtarian, F., Mackworth, A.: Scale-based description and recognition of planar curves and two-dimensional shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI* 8, 34–43 (1986)
16. Kerautret, B., Lachaud, J.O.: Robust estimation of curvature along digital contours with global optimization. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) DGCI 2008. LNCS, vol. 4992, pp. 334–345. Springer, Heidelberg (2008)
17. Kim, E., Haseyama, M., Kitajima, H.: Fast and Robust Ellipse Extraction from Complicated Images. In: Proceedings of the International Conference on Information Technology and Applications, pp. 357–362 (2002)

Estimation of the Derivatives of a Digital Function with a Convergent Bounded Error

Laurent Provot and Yan Gérard

Univ. Clermont 1, ISIT, Campus des Cézeaux, 63172 Aubière, France
provot.research@gmail.com, yan.gerard@u-clermont1.fr

Abstract. We provide a new method to estimate the derivatives of a digital function by linear programming or other geometrical algorithms. Knowing the digitization of a real continuous function f with a resolution h , this approach provides an approximation of the k^{th} derivative $f^{(k)}(x)$ with a maximal error in $O(h^{\frac{1}{1+k}})$ where the constant depends on an upper bound of the absolute value of the $(k+1)^{\text{th}}$ derivative of f in a neighborhood of x . This convergence rate $\frac{1}{k+1}$ should be compared to the two other methods already providing such uniform convergence results, namely $\frac{1}{3}$ from Lachaud *et. al* (only for the first order derivative) and $(\frac{2}{3})^k$ from Malgouyres *et al.*

Keywords: Derivative estimation, Digital Level Layer, Convergence rate, Linear Programming.

1 Introduction

One of the main goal of digital geometry is to provide a theory of digital differential geometry valid on digital surfaces or digital functions. In this framework, the question of the derivative of a function at any order k is central. To be suitable for applications, we are waiting from a digital derivative that it remains close from a continuous derivative if we apply it on the restriction of the function on a digital subset around a point. More formally, it means that we want a property of uniform convergence if the resolution h tends to 0. Such a property of digital derivatives has been first investigated by Vialard, Lachaud, de Vieilleville, Feschet in [7,4,8] and Brunet, Fourey, Malgouyres, Esbelin in [5,11]. The first approach is based on the estimation of digital tangents and provides a uniform error in $O(h^{1/3})$ for the first order derivative while the second approach dealing with binomial convolutions has a maximal error in $O(h^{(2/3)^k})$ for the derivative of order k . The purpose of the paper is to provide a third method based on Taylor polynomial with a uniform error in $O(h^{\frac{1}{k+1}})$. The drawback of the method is that it depends on a parameter of maximal roughness.

The principle is to fit the values of a digital function by a polynomial. Exact fitting, namely interpolation, is not always possible but we expand the values of the function $F(x)$ in intervals $[F(x) - R; F(x) + R]$ so that we can find a polynomial $P(X)$ verifying $P(x) \in [F(x) - R; F(x) + R]$ in a neighborhood of x_0 . How can we choose R ? and the neighborhood of x_0 ?

We start the paper by introducing the notion of roughness of a function, namely the minimum R necessary to find a polynomial approximating the function F with an error smaller than R . We then use this polynomial approximation P to provide derivatives of F . In Section 4, we present theoretical results by bounding the error and in Section 5, we provide some experimental results.

2 Roughness of Order k

2.1 Definition

Let $F : X \rightarrow \mathbb{R}$ be a real function defined on a domain X which can be any subset of \mathbb{R} , but most often a real interval, \mathbb{Z} or a finite subset of consecutive integers. A classic idea about any function f is to approximate its values by a polynomial $P(X) \in \mathbb{R}^k[X]$ of bounded degree. Except in degenerated cases or if the cardinality of X is less than or equal to $k + 1$, it is not possible to find a polynomial in $\mathbb{R}^k[X]$ fitting exactly F , namely satisfying for all x in X the equalities $F(x) = P(x)$. Hence approximating methods have been developed, such as Least Squares Fitting. In the framework of this paper, we focus on uniform fitting:

Definition 1. For any function $F : X \rightarrow \mathbb{R}$ with $X \subset \mathbb{R}$ and any fixed order k , the roughness of F of order k , denoted $Roughness_k(F)$ is the lower bound of the values R such that there exists a polynomial $P(X)$ in $\mathbb{R}^k[X]$ fitting F with a uniform error smaller than R , namely

$$Roughness_k(F) = \inf_{P \in \mathbb{R}^k[X]} \left(\sup_{x \in X} |P(x) - F(x)| \right).$$

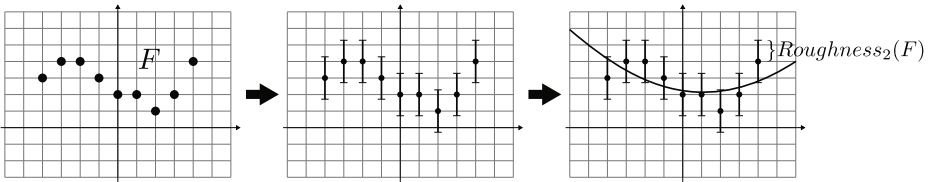


Fig. 1. On the left, a function F . In the middle, we expand each value of F in an interval as small as possible so that there exists a polynomial of degree at most 2 passing through each vertical segment (drawn on the right). This minimal radius around the values of F is the roughness of order 2 of F since we use here polynomials of degree at most 2.

What does it mean in practice? That instead of considering the exact value $F(x)$ at each x of X , we expand it in an interval $I_x = [F(x) - R; F(x) + R]$. There is a threshold $R_k(F)$ such that, with $R < R_k(F)$, there exists no polynomial of degree at most k verifying $P(x) \in I_x$ for any $x \in X$ while for $R > R_k(F)$ there exists at least one polynomial satisfying these conditions (Fig. 1). This polynomial is used in next sections to provide an estimation of the derivative of order k of F .

We can also notice that if X is finite, the lower bound becomes a minimum. If X is not bounded, it is possible to find an infinite roughness, for instance if F is an exponential, its difference with a polynomial can not be bounded. Last, we add that the sequence $Roughness_k(F)$ is decreasing with k and decreases until 0 if X is finite. We have $R_k(F) = 0$ when $k \geq |X| - 1$ (a null roughness means that there exists an exact fitting of F by a polynomial).

2.2 Taylor-Lagrange Inequality

Let us recall Taylor-Lagrange inequality for a function $F : \mathbb{R} \rightarrow \mathbb{R}$ of class C^{k+1} . We consider the Taylor Polynomial $T_{x_0}(x) = \sum_{i=0}^k \frac{f^{(i)}(x_0)}{i!} (x - x_0)^i$ of degree k . If we assume that $|f^{(k+1)}(x)| \leq M$ for $x \in [x_0 - r; x_0 + r]$, we have the inequality $|F(x) - T_{x_0}(x)| \leq M \frac{(x-x_0)^{k+1}}{(k+1)!}$. It means that for any x in the interval $[x_0 - r; x_0 + r]$, the uniform distance between the Taylor polynomial at x_0 and F is less than or equal to $M \frac{r^{k+1}}{(k+1)!}$. By taking $x_0 = \frac{a+b}{2}$ and $2r = b - a$, it leads to the next property:

Property 1. The roughness of the restriction $F_{[a;b]}$ of a function F on an interval $[a; b]$ is smaller than $\|F_{[a;b]}^{(k+1)}\|_{\infty} \frac{(\frac{b-a}{2})^{k+1}}{(k+1)!}$.

2.3 Roughness and Vertical Thickness

The roughness of order 1 is related to the vertical thickness of the graph of F but it is not only true for the order 1. Let us recall the definition of the thickness: Given a finite subset S of \mathbb{R}^d (we call *vertical* the direction of the last coordinate), we define its vertical thickness as the minimal value δ such that all points x of S verify $h \leq n.x \leq h + \delta$ where the last coordinate of the normal vector n is 1. The double inequality $h \leq n.x \leq h + \delta$ simply means that the point x is between the hyperplane of equation $h = n.x$ and its translation by vector $(0, \dots, 0, \delta)$. Hence the vertical thickness is the minimal vertical height of a strip containing S between two parallel hyperplanes (Fig. 2).

Now if we look at the double inequalities defining the roughness of order k of F , we find exactly the same kind of constraints: By denoting a_i the coefficients of polynomial $P(X)$, the double inequality $F(x) - R \leq P(x) \leq F(x) + R$ can be rewritten $a_0 - R \leq -\sum_{i=1}^k a_i x^i + 1F(x) \leq a_0 + R$. This is exactly $h \leq n.x \leq h + \delta$

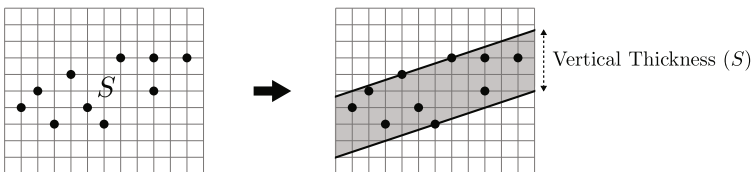


Fig. 2. On the left, a finite set S . On the right, the vertical thickness of S is the minimal height of a strip containing S .

where h plays the role of $a_0 - R$, where δ is equal to $2R$, where the unknown normal vector n (of last coordinate 1) corresponds to the opposite of the unknown coefficients of the polynomial completed by a last coordinate equal to 1 namely $n = ((-a_i)_{1 \leq i \leq k}; 1)$ and where the input points x are given by $((x^i)_{1 \leq i \leq k}; F(x))$ for all x in X . It leads to the property:

Property 2. The roughness of order k of F is the half of the vertical thickness of the set $S = \{((x^i)_{1 \leq i \leq k}; F(x)) \text{ for } x \in X\}$.

2.4 Computation

The roughness can be computed by Linear Programming or directly by some algorithms of Computational Geometry.

Linear Programming. For computations, we consider a function $F : X \rightarrow \mathbb{R}$ defined on a finite domain X . By denoting a_i the coefficients of the fitting polynomial $P(X) = \sum_{i=1}^k a_i X^i$, the roughness of order k of F is the minimum of the objective function R under linear constraints $F(x) - R \leq \sum_{i=1}^k a_i x^i \leq F(x) + R$ for any $x \in X$ (variables are coefficients a_i and R while the values x and $F(x)$ are given in the input). Hence the minimum of R can be computed by any algorithm of linear programming (Simplex, Interior Points, ...). It means that its value can be obtained in linear time in the size of X since linear programming in fixed dimension can be solved in $O(n)$ time where n is the number of linear constraints [6].

Computational Geometry. According to property 2, we can compute the roughness as the vertical thickness of a set in dimension $k + 1$. The first idea to compute this thickness is to pass through the convex hull of the set of points. There is however another faster approach based on the chords set: the vertical thickness of the set S can be obtained as the height of the facet of the convex hull of $S - S$ (the chords) cutting the vertical axis (see [2] for details). The principle of the chords algorithm is to climb all along the vertical axis with the same kind of ideas used in QuickHull or GJK [3]. The time of computation is quasi-linear in practice but its generalization in nD requires new ideas in order to avoid loops.

3 Digital Estimation of $F^{(k)}(0)$

3.1 Parameter

Our estimation of the order k derivative of a function F defined on a discrete domain X at a given point is a direct corollary of the previous notion of roughness. For convenience, we consider a function $F : \mathbb{Z} \rightarrow \mathbb{Z}$. The aim of this section is to provide a method to estimate a kind of k^{th} derivative of F at 0. By translation, it provides a method to estimate the derivatives $F^{(k)}(x)$ at any integer. Unfortunately, we do not provide a canonical definition of these values. The estimation method we provide depends on a parameter of maximal roughness

R_{max} . We will see in the next section that by choosing this parameter greater than $\frac{1}{2} + \frac{\|f^{(k+1)}(x)\|_\infty}{(k+1)!}$, we obtain nice results of convergence of the digital derivative of maximal roughness R_{max} towards the real continuous derivative.

It leads of course to the question: can the maximal roughness be chosen automatically? Previous remark relates it to an upper bound of the $k + 1$ -derivative around the considered point. Hence a possible maximal roughness R_{max} can be estimated with finite differences (for instance $\frac{F(1)-F(-1)}{2}$ for an estimation of $F'(0)$, $F(1) + F(-1) - 2F(0)$ for an estimation of $F''(0)$, ...). As we will see in experiments, another approach is to fix *a priori* this threshold between $\frac{1}{2}$ and 1 ($\frac{1}{2}$ seems to be a minimum to take into account the gaps of the digitization).

3.2 Definition

We consider the restriction $F_{[-m;m]}$ of the function F in a neighborhood of 0 going from $-m$ to m . Given the maximal roughness parameter R_{max} , if m is small enough, the roughness of order k of $F_{[-m;m]}$ is smaller than this maximum authorized R_{max} (if $m = 0$, the roughness at any order k is null). The idea is to increase m until the roughness of $F_{[-m;m]}$ becomes greater than the fixed parameter R_{max} . The intervals $[-m; m]$ providing a roughness smaller than the limit are called k -neighborhood of 0 wrt. F :

Definition 2. *Given $F : \mathbb{Z} \rightarrow \mathbb{Z}$ and a maximal roughness R_{max} , a k -neighborhood of 0 is an integer interval $[-m; m]$ such that the restriction $F_{[-m;m]}$ has a roughness of order k less than R_{max} . The maximal k -neighborhood $[-m_{max}; m_{max}]$ of 0 is the largest k -neighborhood.*

There are two versions of these definitions, one with a strict inequality $Roughness_k(F_{[-m;m]}) < R_{max}$ and another one where we allow a large inequality $Roughness_k(F_{[-m;m]}) \leq R_{max}$. At this step, we are not enough advanced to determine if one is better than the other. We can just notice that if we want the notion of maximal neighborhood of order 1 to collapse with the notion of digital tangent already defined in digital geometry [7], we should choose $R_{max} = \frac{1}{2}$ and a strict inequality since digital straight segments have a vertical thickness strictly less than 1. This leads to the introduction of the derivative of order k :

Definition 3. *Let $[-m; m]$ be the maximal k -neighborhood of 0 for a function F of maximal roughness R_{max} . There exists at least a polynomial $P(X) = \sum_{i=1}^k a_i x^i$ in $\mathbb{R}^k[X]$ verifying $F(x) - Roughness_k(F_{[-m;m]}) \leq P(x) \leq F(x) + Roughness_k(F_{[-m;m]})$ for any integer x from $-m$ to m . We define the k^{th} derivative of F at 0 by $F^{(k)}(0) = k!a_k$.*

Attentive readers will notice that it can arise non unique solutions $P(X)$ in degenerated cases. In this case, which coefficient a_k should we choose to define the derivative? Again, depending on the application we can leave open the choice, consider that there is no good value and that it is better to provide an interval as a solution, or that the mean on the interval $[a_{k_{min}}; a_{k_{max}}]$ is the best representative of the possible derivatives that can be chosen.

Of course, this definition does not match all the properties of continuous derivative. We have for instance no guarantee that by derivating a derivative of order k , we obtain the same value as by computing directly the derivative of order $k+1$. Hence how can we say that this definition is suitable? with a property we investigate in the next section in order to show that the digital derivative estimation converges towards the continuous value.

4 Error Bounding

Let us consider the framework of Numerical Analysis or Signal Theory. We consider a real function $f : \mathbb{R} \rightarrow \mathbb{R}$ of class C^{k+1} known from its digitization in a grid of horizontal and vertical resolution h . It means that we know a portion of the digital function $F : \mathbb{Z} \rightarrow \mathbb{Z}$ defined by $F(x) = \lfloor \frac{f(hx)}{h} \rfloor$. We can estimate the derivatives of F around 0 with a given maximal roughness by the method described in the previous section. As we have here the initial value of f , the question is to determine the maximal error of this method of estimation on the real derivative of order k –corrected by a scale factor h^{1-k} – namely to bound the difference $f^{(k)}(0) - h^{1-k}F^{(k)}(0)$. This scale factor comes from the fact that the derivative of order k of $x \mapsto \frac{f(hx)}{h}$ is $(\frac{f(hx)}{h})^{(k)} = h^{k-1}f^{(k)}(hx)$. We provide the following upper bound:

Theorem 1. *Let $f : \mathbb{R} \rightarrow \mathbb{R}$ be a real function of class C^{k+1} whose derivative of order $k + 1$ is bounded by M in a neighborhood of 0 (more precisely we assume $|f^{(k+1)}(x)| \leq M$ for $x \in [-L; L]$). We have $|f^{(k)}(0) - h^{1-k}F^{(k)}(0)| \leq u_k(\frac{1}{2} + \frac{M}{(k+1)!} + R_{max})h^{\frac{1}{k+1}}$ for the estimation of $F^{(k)}(0)$ with roughness $R_{max} \geq \frac{1}{2} + \frac{M}{(k+1)!}$.*

The proof relies on two inequalities: the first one says that the difference between Taylor polynomial $T(x)$ of $\frac{f(hx)}{h}$ around 0 of order k and $F(x) = \lfloor \frac{f(hx)}{h} \rfloor$ is small. The second one expresses the fact that the difference between $\lfloor \frac{f(hx)}{h} \rfloor$ and the polynomial $P(x)$ that coefficients are obtained by computations of the approximation F with maximal roughness R_{max} is also small in a discrete neighborhood of 0. It leads to bound the difference between $P(x)$ and $T(x)$ ($\lfloor \frac{f(hx)}{h} \rfloor$ plays the role of intermediary value) on a discrete neighborhood. With an important lemma on discrete norms on polynomials, we obtain the claimed result. Details of the proof are given in next subsections.

4.1 Bounding the Difference $T(x) - P(x)$ on a Discrete Neighborhood

It is the first step of the proof of Theorem **1**. We consider values of x in the interval $[-h^{-\frac{k}{k+1}}; h^{-\frac{k}{k+1}}]$ with h small enough to have hx included in $[-L; L]$ for all these values (in other words h is chosen smaller than L^{k+1}). It follows that the $(k + 1)^{th}$ derivative of $\frac{f(hx)}{h}$ namely $h^k f^{(k+1)}(hx)$ verifies $(\frac{f(hx)}{h})^{(k+1)} \leq h^k M$.

According to Taylor-Lagrange inequality at point 0 for function $\frac{f(hx)}{h}$, we have $|T(x) - \frac{f(hx)}{h}| \leq \frac{h^k M}{(k+1)!} x^{k+1} \leq \frac{h^k M}{(k+1)!} (h^{-\frac{k}{k+1}})^{k+1}$ namely

$$\left| T(x) - \frac{f(hx)}{h} \right| \leq \frac{M}{(k+1)!} \tag{1}$$

where $T(x)$ is the Taylor polynomial of $\frac{f(hx)}{h}$ at 0: $T(x) = f(0) + \sum_{i=1}^k h^{i-1} \frac{f^{(i)}(0)}{i!} x^i$. As we have $0 \leq \frac{f(hx)}{h} - \lfloor \frac{f(hx)}{h} \rfloor \leq 1$, it follows that $\left| \lfloor \frac{f(hx)}{h} \rfloor + \frac{1}{2} - \frac{f(hx)}{h} \right| \leq \frac{1}{2}$ and with bounding (1):

$$\forall x \in \left[-h^{-\frac{k}{k+1}}; h^{-\frac{k}{k+1}} \right], \quad \left| T(x) - \frac{1}{2} - \left\lfloor \frac{f(hx)}{h} \right\rfloor \right| \leq \frac{1}{2} + \frac{M}{(k+1)!} \tag{2}$$

Now we can be interested in the bounding of the difference between $P(x)$ and $F(x) = \lfloor \frac{f(hx)}{h} \rfloor$. According to inequality (2) and condition $R_{max} \geq \frac{1}{2} + \frac{M}{(k+1)!}$ of Theorem 1, it follows that the interval $[-h^{-\frac{k}{k+1}}; h^{-\frac{k}{k+1}}]$ is a neighborhood of 0 for maximal roughness R_{max} . Hence the maximal neighborhood of 0 for the function F is at least the interval $[-h^{-\frac{k}{k+1}}; h^{-\frac{k}{k+1}}]$.

Now let us introduce the polynomial $P(x)$ obtained with our method to estimate the derivative of a digital function : it is a solution of inequalities $|P(x) - F(x)| \leq Roughness_k(F_{[-m;m]})$ where $[-m;m]$ is the maximal k -neighborhood of 0 for the function F and maximal roughness R_{max} . As the maximal neighborhood contains $[-h^{-\frac{k}{k+1}}; h^{-\frac{k}{k+1}}]$, we have:

$$\forall x \in [-h^{-\frac{k}{k+1}}; h^{-\frac{k}{k+1}}] \cap \mathbb{Z}, \quad \left| P(x) - \left\lfloor \frac{f(hx)}{h} \right\rfloor \right| \leq R_{max} \tag{3}$$

It follows directly from inequalities (2) and (3)

$$\forall x \in [-h^{-\frac{k}{k+1}}; h^{-\frac{k}{k+1}}] \cap \mathbb{Z}, \quad \left| T(x) - \frac{1}{2} - P(x) \right| \leq \frac{1}{2} + \frac{M}{(k+1)!} + R_{max} \tag{4}$$

4.2 Discrete Norms on Polynomials

The aim of this subsection is to show that two polynomials whose difference can be bounded on a discrete interval $\{-m\dots m\}$ – as previous polynomials $T - \frac{1}{2}$ and P – have necessarily close coefficients.

Lemma 1. *We assume $k < 2m$. Let $A(X)$ and $B(X)$ be two polynomials of degree at most k whose difference $|A(X) - B(X)|$ is less than a constant C on a discrete interval $\{-m\dots m\}$. We have a difference $|a_k - b_k|$ of coefficients of degree k which is smaller than $Cv_k m^{-k}$ where v_k is a constant depending only on k .*

Let us denote $\alpha_k(m, n)$ the maximum of the coefficient of degree k among all polynomials $P(X)$ of degree at most n and verifying $|P(x)| \leq 1$ for all integers x from $-m$ to m .

Property 3. – The value $\alpha_k(m, n)$ decreases with m .

- The value $\alpha_k(m, n)$ increases with n .
- For any positive integer μ , we have $\alpha_k(\mu m, n) \leq \alpha_k(m, n)/\mu^k$.
- We have $1/m^k \leq \alpha_k(m, n)$
- If we assume $m \geq 2p$ where $p = \lfloor (n - 1)/2 \rfloor + 1$, we have $\alpha_k(m, n) \leq (2p)^k \alpha_k(p, n)/m^k$ namely $m^k \alpha_k(m, n)$ is bounded as a function of m .

Proof. The decrease of $\alpha_k(m, n)$ in m and increase in n are straightforward.

For going from $\alpha_k(\mu m, n)$ to $\alpha_k(m, n)$, we just have to notice that if $P(X)$ satisfies $-1 \leq P(x) \leq 1$ for any integer x from $-\mu m$ to μm , then $P(\mu X)$ whose coefficient of degree k is $\mu^k a_k$ verifies $-1 \leq P(\mu x) \leq 1$ for all integers x from $-m$ to m . It follows that $|\mu^k a_k| \leq \alpha_k(m, n)$. It proves that the maximum of $|a_k|$ is at most $\alpha_k(m, n)/\mu^k$ namely $\alpha_k(\mu m, n) \leq \alpha_k(m, n)/\mu^k$.

To prove $1/m^k \leq \alpha_k(m, n)$, we just have to notice that the polynomial $P(X) = X^k/m^k$ verifies $-1 \leq P(x) \leq 1$ for all integer x from $-m$ to m . Hence the maximum $\alpha_k(m, n)$ of the coefficient of degree k under these constraints is at least $1/m^k$.

For the last inequality, we define $p = \lfloor (n - 1)/2 \rfloor + 1$ because it is the first integer for which $\alpha_k(p, n)$ is defined (it verifies $2p + 1 > n$). The assumption $m \geq 2p$ allows to write $m \leq 2m - 2p$ which leads to (i) $m/p \leq 2(m - p)/p = 2(m/p - 1)$. As the function $\alpha_k(m, n)$ increases with m , we have $\alpha_k(m, n) \leq \alpha_k(\lfloor m/p \rfloor p, n)$. With $\mu = \lfloor m/p \rfloor$, it provides (ii) $\alpha_k(m, n) \leq \alpha_k(p, n)/\lfloor m/p \rfloor^k$. If we look at the denominator $\lfloor m/p \rfloor^k$, we have $\lfloor m/p \rfloor > m/p - 1 \geq m/2p$ according to (i). Hence we obtain $\lfloor m/p \rfloor^k > (m/2p)^k$. In our main inequality (ii), it leads to $\alpha_k(m, n) \leq (2p)^k \alpha_k(p, n)/m^k$.

The property 3 shows that the values $\alpha_k(m, n)$ decrease in $1/m^k$. Lemma 1 is just a consequence of this property for $n = k$, i.e $\alpha_k(m, k) \leq \frac{v_k}{m^k}$.

4.3 Some Values for $\alpha_k(m, n)$

- With $n = 0$, we have $\alpha_0(m, 0) = 1$. More generally we have $\alpha_0(m, n) = 1$.
- With $n = 1$, we have $\alpha_1(m, 1) = 1/m$.
- With $n = 2$, we have $\alpha_1(m, 2) = 1/m$ and $\alpha_2(m, 2) = 2/m^2$.
- With $n = 3$, we have $\alpha_1(m, 3) = \frac{3}{m}$ if m is even and $\alpha_1(m, 3) = \frac{3m^2 + 1}{(m - 1)m(m + 1)}$ if m is odd. $\alpha_2(m, 3) = 2/m^2$. $\alpha_3(m, 3) = \frac{4}{m^3}$ if m is even and $\alpha_3(m, 3) = \frac{4}{(m - 1)m(m + 1)}$ if m is odd.
- With $n = 4$, we have $\alpha_1(m, 4) = \alpha_1(m, 3)$, $\alpha_2(m, 4) = \frac{2m^2}{(m - i)(m + i)i^2}$, $\alpha_3(m, 4) = \alpha_3(m, 3)$ and $\alpha_4(m, 4) = \frac{2}{(m - i)(m + i)i^2}$ where i is the closest integer from $\frac{m}{\sqrt{2}}$.

4.4 Synthesis

To obtain Theorem 1, we just apply Lemma 1 on inequality (3) with $T(X) - \frac{1}{2}$ and $P(X)$ as polynomials $A(X)$ and $B(X)$, $\frac{1}{2} + \frac{M}{(k + 1)!} + R_{max}$ as constant C and $m = \lfloor h^{-\frac{k}{k + 1}} \rfloor$. We first recall that our estimation of the derivative $F^{(k)}(0)$ is $k!a_k$

where a_k is the coefficient of degree k of P . For $k > 0$, the coefficient of degree k of $T(X) - \frac{1}{2}$ is $h^{k-1} \frac{f^{(k)}(0)}{k!}$. It follows $\left| \frac{F^{(k)}(0)}{k!} - h^{k-1} \frac{f^{(k)}(0)}{k!} \right| \leq \left(\frac{1}{2} + \frac{M}{(k+1)!} + R_{max} \right) \frac{v_k}{m^k}$ namely $\left| h^{1-k} \frac{F^{(k)}(0)}{k!} - f^{(k)}(0) \right| \leq \left(\frac{1}{2} + \frac{M}{(k+1)!} + R_{max} \right) k! v_k \frac{h^{1-k}}{m^k}$. It remains to say that for h small enough (according to k) we have $m = \lfloor h^{-\frac{k}{k+1}} \rfloor \geq (1 - \epsilon) h^{-\frac{k}{k+1}}$ and thus $m^{-k} \leq (1 - \epsilon)^{-k} m^{\frac{k^2}{k+1}}$. It leads to $\left| h^{1-k} \frac{F^{(k)}(0)}{k!} - f^{(k)}(0) \right| \leq \left(\frac{1}{2} + \frac{M}{(k+1)!} + R_{max} \right) k! (1 - \epsilon)^{-k} v_k h^{1-k + \frac{k^2}{k+1}}$. With $u_k = k! (1 - \epsilon)^{-k} v_k$ it provides Theorem \square .

4.5 State of the Art

The previous approach based on Taylor Polynomial provides a numerical algorithm to estimate the derivatives of a function known from its digitization with a guaranteed result. It is however not the first method to solve this problem. Several works have been developed in order to estimate the derivatives of such a function. Lachaud *et al.* provide a method with a mean convergence rate in $O(h^{2/3})$ and a worst case convergence rate in $O(h^{1/3})$ to estimate the tangent of the curve, namely its first order derivative [4]. This approach is based on maximal digital segment recognition and is thus rather close to the approach provided here (the difference is in the choice of the maximal thickness or roughness). The construction of a curvature estimator that has the property of asymptotic convergence was however still open in this framework [8]. More recently, Malgouyres *et al.* provide a new method based on binomial estimation allowing to estimate the derivatives at any order [5][1]. The maximal error of this k^{th} derivative estimator converges in $O(h^{(2/3)^k})$.

It makes at least two other methods to estimate the derivatives of a function known from its digitization with a controlled uniform error. As far as we know, no other result has been published in this specific framework. Hence there are three methods to compare according to at least three theoretical criteria:

The speed of convergence: The speeds of convergence of the three methods are of the form $O(h^d)$ where h is the resolution (tending to 0) and d is the degree. The higher the exponent d , the better the convergence. For the method based on discrete segments, we have a degree (in the worst case) of $\frac{1}{3}$ only for the first order derivative. For the binomial method the degree of convergence is $(\frac{2}{3})^k$ to estimate the derivative of order k while the approach based on Taylor Polynomial and Linear Programming provides a degree of $\frac{1}{k+1}$. We compare these three degrees for the orders of derivatives from 1 to 5 (Tab. \square).

Table 1. Comparison between the degree of convergence of the three methods to estimate the k^{th} derivative

k	1	2	3	4	5
	$1/3 = 0.333$				
$(\frac{2}{3})^k$	$2/3 = 0.666$	$4/9 = 0.444$	$8/27 = 0.296$	$16/81 = 0.19753$	$32/243 = 0.1317$
$\frac{1}{k+1}$	$1/2 = 0.5$	$1/3 = 0.333$	$1/4 = 0.25$	$1/5 = 0.2$	$1/6 = 0.1666$

The number of points necessary to compute the estimation: For the approach based on maximal digital segments, the number of points is the size of the maximal segment which has no upper bound (there is an upper bound for the smallest digital segment in $\Omega(m^{1/3} \log m)$ in [8]). With the convolution method, the first value given for the size of the mask is $m = h^{-\frac{1}{1.01}}$ but a proposition allows to work with only $O(\sqrt{m \ln m})$ points [1]. With the Taylor Polynomial approach, the relation $h = m^{-1-\frac{1}{k}}$ leads to use at least $2m + 1 = 2h^{-\frac{k}{k+1}} + 1$ points. This number of points depends on the order of the derivative. For $k = 1$, the degree is $-\frac{1}{2}$. For $k = 2$, it is $-\frac{2}{3}$ and more generally $-\frac{k}{k+1}$ for the derivative of order k (we need more points if we increase k).

The complexity of the computation: In both cases, the computation time is directly related to the number of points, i.e. to the previous criterion. For the computation of the maximal digital segment, this can be done in linear time in the number of points. For the convolution, the time of computation is again linear in the size of the binomial mask m . For the Taylor Polynomial, the method uses a linear program. Hence the time of computation could be linear in m ($O(m)$ with Megiddo algorithm [6]). But without any information on the minimal neighborhood that can be chosen to provide an approximation, we have to increase the neighborhood at each step and check that the maximal roughness is not exceeded. This process can be improved – as in the framework of digital tangents – but this naive approach provides a time of computation quadratic in the size of the maximal neighborhood.

We have compared the three approaches according to theoretical criteria but it would be probably more enlightening to compare them on a benchmark of functions whose derivative are known.

5 Experiments

We propose in this section some results obtained with our Taylor based method to estimate derivatives of a digital function $F(x)$. Fig. 3 shows a digitization

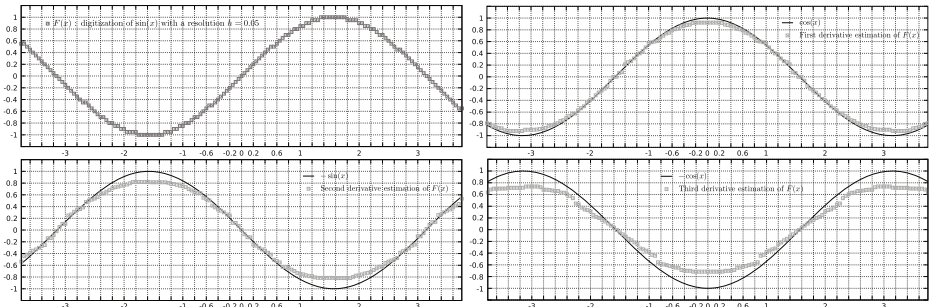


Fig. 3. A digitization $F(x)$ of the sine function at resolution $h = 0.05$ and an estimation of its derivatives from order 1 to 3 by using our Taylor based method

of the sine function (according to the grid intersect quantization (GIQ), i.e. $F(x) = \lfloor \frac{\sin(hx)}{h} + \frac{1}{2} \rfloor$) with a resolution $h = 0.05$ and an estimation of its first, second and third derivatives with a roughness $R_{max} = \frac{1}{2}$. The continuous derivative has been drawn for each estimation to compare with the expected theoretical results. To emphasize the influence of the resolution, the sine function has been digitized (GIQ) with different values of h and an estimation of the second derivative of this digitization (with $R_{max} = \frac{1}{2}$) has been done. The results are depicted in Fig. 4. As we might expect the finer the resolution is, the more

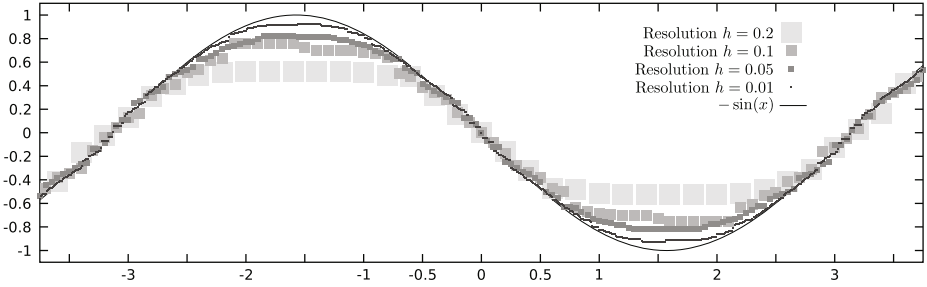


Fig. 4. Estimations of the second derivative of the digitization of sine at different resolutions

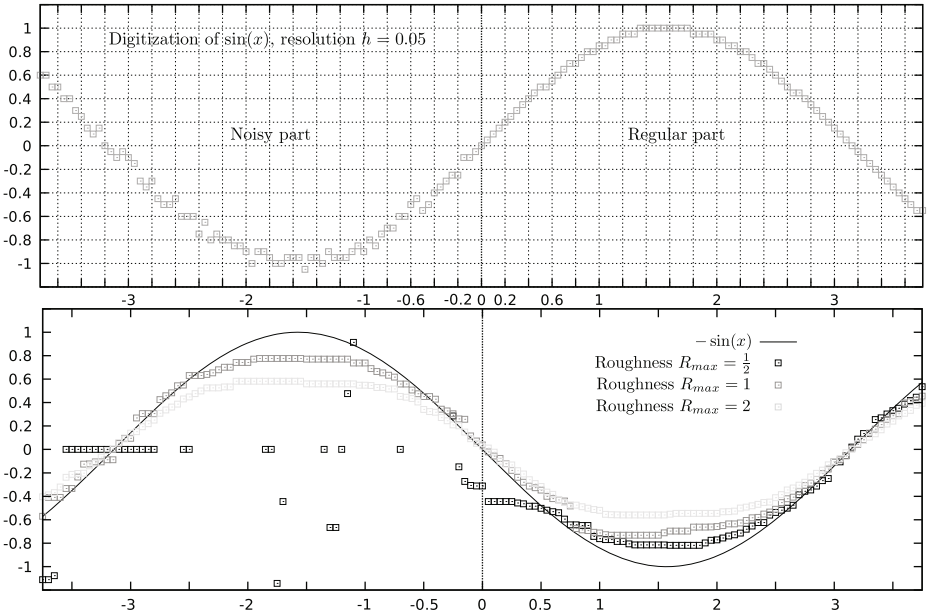


Fig. 5. Estimations of the second derivative of the digitization of sine at different roughness. The left part ($x < 0$) of the digitization is noisy to see the influence of the roughness parameter.

accurate the estimation is. Last, an estimation of the second derivative of a digitization of the sine with a resolution $h = 0.05$ has been computed at different roughness (Fig. 5). To see the effect of this parameter, the digitization has been partially perturbed (when $x < 0$) with a uniform noise. Although the most accurate results on the regular part of the digitization are obtained with the smallest roughness $R_{max} = \frac{1}{2}$, on the noisy part the estimation is completely distorted. If we increase the roughness the estimation becomes reliable in the noisy part, but higher roughness values tend to smooth the results.

6 Conclusion

We have proposed in this paper a method to estimate the derivatives of a digital function. It is a kind of generalization of the method based on digital tangents. The main idea was to relax the thickness constraint, our roughness parameter, and to allow the degree of the fitting polynomial to be greater than 1. This led to an interesting theoretical result of convergence in $O(h^{\frac{1}{k+1}})$. A perspective would be to extend this framework to multivariable functions but much more experiments are required to determine the relative benefits or drawbacks of this method, in terms of time of computations and accuracy of the results, compared to the convolutions approach.

References

1. Esbelin, H.-A., Malgouyres, R.: Convergence of binomial-based derivative estimation for C^2 noisy discretized curves. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 57–66. Springer, Heidelberg (2009)
2. Gerard, Y., Debled-Rennesson, I., Zimmermann, P.: An elementary digital plane recognition algorithm. *Discrete Applied Mathematics* 151, 169–183 (2005)
3. Gilbert, E.G., Johnson, D.W., Keerthi, S.S.: A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal of Robotics and Automation* 4, 193–203 (1988)
4. Lachaud, J.O., Vialard, A., de Vieilleville, F.: Fast, accurate and convergent tangent estimation on digital contours. *Image and Vision Computing* 25(10), 1572–1587 (2007)
5. Malgouyres, R., Brunet, F., Fourey, S.: Binomial convolutions and derivatives estimation from noisy discretizations. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) DGCI 2008. LNCS, vol. 4992, pp. 370–379. Springer, Heidelberg (2008)
6. Megiddo, N.: Linear programming in linear time when the dimension is fixed. *Journal of the ACM* 31(1), 114–127 (1984)
7. Vialard, A.: Geometrical parameters extraction from discrete paths. In: Miguet, S., Ubéda, S., Montanvert, A. (eds.) DGCI 1996. LNCS, vol. 1176, pp. 24–35. Springer, Heidelberg (1996)
8. de Vieilleville, F., Lachaud, J.O., Feschet, F.: Maximal digital straight segments and convergence of discrete geometric estimators. In: Kalviainen, H., Parkkinen, J., Kaarna, A. (eds.) SCIA 2005. LNCS, vol. 3540, pp. 988–997. Springer, Heidelberg (2005)

Properties and Applications of the Simplified Generalized Perpendicular Bisector

Aurélie Richard, Gaëlle Largeteau-Skapin, Marc Rodríguez, Eric Andres, Laurent Fuchs, and Jean-Serge Dimitri Ouattara

Laboratory XLIM, SIC Department,
University of Poitiers BP 30179, UMR CNRS 6712
86962 Futuroscope Chasseneuil Cedex, France

Abstract. This paper deals with the Simplified Generalized Perpendicular Bisector (SGPB) presented in [15,1]. The SGPB has some interesting properties that we explore. We show in particular that the SGPB can be used for the recognition and exhaustive parameter estimation of noisy discrete circles. A second application we are considering is the error estimation for a class of rotation reconstruction algorithms.

Keywords: Simplified Generalized Perpendicular Bisector - Adaptive Pixel Size - Generalized Reflection Symmetry - Rotation Reconstruction.

1 Introduction

In this paper we are discussing properties and applications of the Simplified Generalized Perpendicular Bisector (SGPB) that has been introduced in [15,1]. The SGPB is an extension of the classical notion of perpendicular bisector. The Generalized Perpendicular Bisector (GPB) for two 2D regions A and B is defined as the union of the perpendicular bisectors of all the couple of points (p, q) where p and q are respectively points of the regions A and B . The regions we are going to focus on are pixels. The boundary of a GPB between two pixels is composed of line segments and parabola segments. In order to simplify the computational aspects, the SGPB has been introduced. It is only composed of straight line segments. Contrary to a 2D perpendicular bisector, the GPB and SGPB are not lines but surfaces. Bisectors between points and curves or between two curves have also been discussed in detail in the literature [7,11] but to our best knowledge, no definition for the bisector between two coplanar surfaces such as pixels has been proposed before [15].

The SGPB has some interesting properties and can be useful in several different applications mainly when noisy data is considered. The first application we present is the adaptive pixel SGPB and its use for noisy circle recognition. This is similar to the idea presented in [14] for discrete straight lines. The second application concerns the parameter estimation of rotations. For this, the Generalized Perpendicular Symmetry transform is introduced. We use this to examine the rotation estimation algorithm recently proposed by Fontjine et al. [8] and illustrate its behaviour for noisy data.

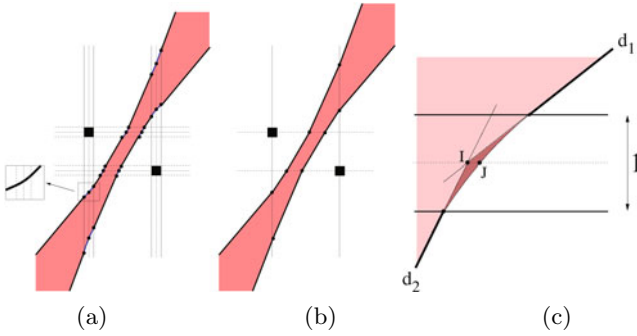


Fig. 1. The exact perpendicular bisector (a) and the approximation (b) where the parabolic pieces have been dropped by extending the straight lines (c). The generalized bisector is slightly reduced.

The paper is organized as follows: section 2 recalls some properties of the SGPB. The section 3 is devoted to the SGBP and the adaptive pixel size SGBP. In particular some of its properties are highlighted and an application to noisy circle recognition is provided. The next section deals with the symmetry relatively to a SGPB and its application to rotation parameter estimation.

2 Definition and Properties of the SGBP [15,1]

As mentioned in the introduction, the Generalized Perpendicular Bisector (GPB) between two 2D regions A and B (see Figure 1(a)) is defined as the union of the perpendicular bisectors of all the couples of points (p, q) where p (resp. q) belongs to the regions A (resp. B). In our case we are considering bounded connected regions since we focus on regions that are pixels (that may have different sizes).

By definition, points of the perpendicular bisector are equidistant to both points p and q . Thus a particular point r on the perpendicular bisector is the center of a hypersphere that passes through p and q . Hence, we can set an alternative definition of the GPB. Let \mathcal{S}_1 and \mathcal{S}_2 be two bounded connected regions and X an Euclidean point of \mathbb{R}^n . Let $d_{i_{min}}(X) = \min_{Y \in \mathcal{S}_i}(d(X, Y))$, $d_{i_{max}}(X) = \max_{Y \in \mathcal{S}_i}(d(X, Y))$ where d is the usual Euclidean distance. Every Euclidean point $X \in \mathbb{R}^n$ such that:

$$[d_{1_{min}}(X), d_{1_{max}}(X)] \cap [d_{2_{min}}(X), d_{2_{max}}(X)] \neq \emptyset \tag{1}$$

belongs to the GPB of \mathcal{S}_1 and \mathcal{S}_2 (see Figure 1(a)). The boundary of the GPB between two pixels is composed of line segments and parabola segments. The parabola segments can be easily removed by extending the line segments. This defines the Simplified GPB (SGPB) (See Figure 1(b,c)). Another way of considering this is simply to state that the minimum distance to a pixel is approximated by the distance to the closest vertex of the pixel rather than to the closest vertex or edge. In other words, the SGPB of two pixels P_1 and P_2 of size respectively λ_1 and λ_2 is given by the following equations:

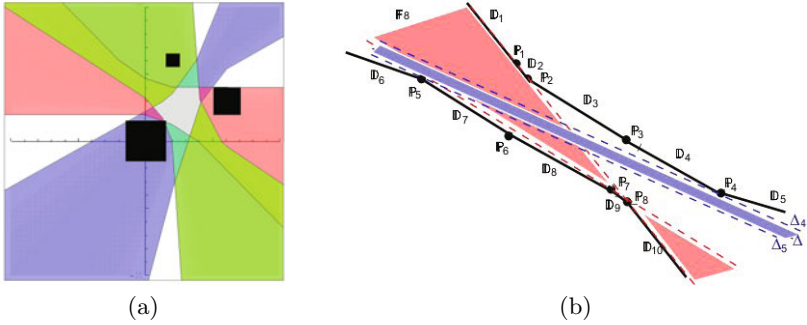


Fig. 2. (a) Simplified Generalized Perpendicular Bisector and Simplified Generalized Circumcenter of three adaptive pixels ; (b) The 6 border segments and the 4 half-lines of a SGPB $\{\mathbb{D}_i\}_{i \in [1,10]}$ and its characteristic points $\{\mathbb{P}_i\}_{i \in [1,8]}$ ($\mathbb{D}_1 = \mathbb{D}_{10}$ and $\mathbb{D}_5 = \mathbb{D}_6$). The line bundle \mathbb{F}_8 and the line beam $beam(\mathbb{P}_4, \mathbb{P}_5)$.

$$SGPB(P_1, P_2) = \left\{ (x, y) \in \mathbb{R}^2, \right. \\ \left. \left(\sqrt{(x - C_{2x})^2 + (y - C_{2y})^2} \leq \sqrt{(x - F_{1x})^2 + (y - F_{1y})^2} \right) \wedge \left(\sqrt{(x - C_{1x})^2 + (y - C_{1y})^2} \leq \sqrt{(x - F_{2x})^2 + (y - F_{2y})^2} \right) \right\}$$

where $C_{ix}, F_{ix} \in \left\{ \left(x_i + \frac{\lambda_i}{2} \right), \left(x_i - \frac{\lambda_i}{2} \right) \right\}$, $C_{iy}, F_{iy} \in \left\{ \left(y_i + \frac{\lambda_i}{2} \right), \left(y_i - \frac{\lambda_i}{2} \right) \right\}$.

The SGPB between two pixels P_1 and P_2 is bounded by line segments and half-lines as shown in the Figure 2(b).

Proposition 1. *The boundary of 2D-Simplified Generalized Perpendicular Bisector between two pixels (x_1, y_1) of size λ_1 and (x_2, y_2) of size λ_2 is composed by at most 10 line segments and half-lines.*

The proof of this proposition is straightforward.

Let us introduce some concepts and notations useful for the purpose of this section. Let $\{\mathbb{D}_i\}_{i \in [1,10]}$ be the border segments and half-lines of $SGPB(P_1, P_2)$. Let $\{\mathbb{P}_i\}_{i \in [1,8]}$ be the characteristic points of $SGPB(P_1, P_2)$: i.e. \mathbb{P}_i belongs to the border of the $SGPB(P_1, P_2)$ and we organize in such a way that $\mathbb{P}_i = \mathbb{D}_i \cap \mathbb{D}_{i+1}$ if $i \in [1; 4]$ and $\mathbb{P}_i = \mathbb{D}_{i+1} \cap \mathbb{D}_{i+2}$ if $i \in [5; 8]$ (see Figure 2(b)). $\mathbb{D}_1, \mathbb{D}_{10}, \mathbb{D}_5, \mathbb{D}_6$ are the support lines for the half-lines on the border of the $SGPB(P_1, P_2)$. In fact, as the following lemma shows half-lines are linked two by two:

A first immediate property is given by following lemma:

Lemma 1. *Let $A(a_1, a_2)$ of size λ_A and $B(b_1, b_2)$ of size λ_B be two pixels such as $a_i \pm \frac{\lambda_A}{2} \neq b_i \pm \frac{\lambda_B}{2}$. The border segments \mathbb{D}_1 and \mathbb{D}_{10} are supported by the same line and this is also the case for segments \mathbb{D}_5 and \mathbb{D}_6 .*

The proof of this lemma is straightforward using formula (1). It is also easy to see that in case $a_i \pm \frac{\lambda_A}{2} = b_i \pm \frac{\lambda_B}{2}$ then \mathbb{D}_1 and \mathbb{D}_{10} or \mathbb{D}_5 and \mathbb{D}_6 are not equal but parallel.

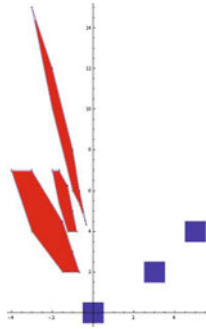


Fig. 3. Dual of three SGPB corresponding to three pixels

2.1 Dual of a SGPB

It is easy to determine if a point belongs to a SGPB but not so for a straight line although the SGPB is defined as a union of straight lines. One way of handling this problem is to consider the dual of a SGPB. The dual (see [6] for more details) of the perpendicular bisector between Euclidean points $A(x_a, y_a)$ and $B(x_b, y_b)$ is defined as the point $(\frac{x_b - x_a}{y_a - y_b}, \frac{x_a^2 - x_b^2 + y_a^2 - y_b^2}{2(y_a - y_b)})$ for $y_a - y_b \neq 0$. In the case where $y_a - y_b = 0$ we have a point at the infinity.

Proposition 2. *The dual of a SGPB is a convex polygon of at most 8 vertices and 8 edges. At most two vertices may be at the infinite and the corresponding edges of the dual polygon are in this case vertical.*

Proof. The boundary of the SGPB is formed of straight lines segments and half-lines. The dual is therefore a polygon. It is easy to see that this polygon is convex. Let us consider two points A and B on the boundary of the dual of a SGPB. These two points correspond to two straight lines L_A and L_B that belong to the SGPB. The intersection point I between L_A and L_B corresponds to the straight line AB in the dual. The point I is inside the SGPB. All the straight lines passing through I form two bundles delimited by L_A and L_B with one of them containing only lines that are inside the SGPB since L_A and L_B are inside the SGPB. The other bundle contains lines that are outside the SGPB. A point outside the segment $[AB]$ and not inside the polygon corresponds to a straight line that does not belong to the SGPB and thus to the bundle that contains lines that do not belong to the SGPB. The points that belong to the segment $[AB]$ correspond therefore to straight lines of the bundle that contain only lines inside the SGPB. All the points of $[AB]$ are therefore inside the polygon and this proves that the polygon is convex. The number of vertices and edges of the polygon is an immediate result of Proposition 1. \square

Determining if a point belongs to a convex polygon with bounded number of edges and thus if a straight line belongs to the SGPB is trivially done in $O(1)$.

3 Adaptive Pixel SGPB and Noisy Circle Recognition

In [14] some of the authors considered a straight line recognition method that takes into account noise by locally increasing the size of the various pixels. Using a resizing function [14] and a local noise estimator (for example the one proposed by Kerautret et al. [10]), pixel sizes are adaptively increased according to the local perturbation on the straight line. The size increased pixels are called *Adaptive Pixels* [14]. The idea here is to do exactly the same with the SGPB for noisy circle recognition purposes.

First, let us introduce the Simplified Generalized Circumcenter (*SGC*) of a set of n finite and connected regions $\mathcal{S} = (S_i)_{i \in [1,n]}$. It is defined as the intersection of the SGPB of every two regions of the set (see Figure 2(a)) :

$$SGC(\mathcal{S}) = \bigcap_{i,j \in [1,n], i < j} (SGPB(S_i, S_j)).$$

Theorem 1. *Each point of the Simplified Generalized Circumcenter corresponds to the center of at least one circle that intersects all the Adaptive Pixels.*

Proof. The proof is similar to the one of Theorem 1 presented in [15]: Each point of the *SGC* obtained by the intersection of adaptive pixel *SGPBs* corresponds to an intersection of radii intervals (dimension 1) for every two adaptive pixels. A direct application of Helly’s Theorem tell us that there exists at least one common radius to all these intervals and thus at least one circle of this radius centered on the *SGC* point that intersects all the adaptive pixels. \square

Proposition 3. *Let us consider a set of adaptive pixels P_i of various sizes. The dual of all the straight lines crossing the duals of all the SGPB of every pair of pixels P_i and P_j is the dual of the Simplified Generalized Circumcenter.*

This proposition is an immediate consequence of the definition of the *SGC* and the definition of our notion of dual.

3.1 Application to Noisy Circle Recognition

On the Figure 4, we can see a Bresenham circle of radius 5 with misplaced and missing pixels. The *SGC* can be seen in the middle of the different adaptive pixels. One circle example (the center is marked by a black dot in the *SGC*) is shown. Each pixel size is increased according to a local noise estimator. The algorithm is the same as the one for regular circles presented in [15]. The *SGPB* of each couple of pixels (with the new sizes) is computed and their intersection provides a set of possible circle centers, the Simplified Generalized Circumcenter (*SGC*). A further computation provides for each point of the *SGC*, the interval of possible circle radii that correspond to circles intersecting all the size increased pixels.

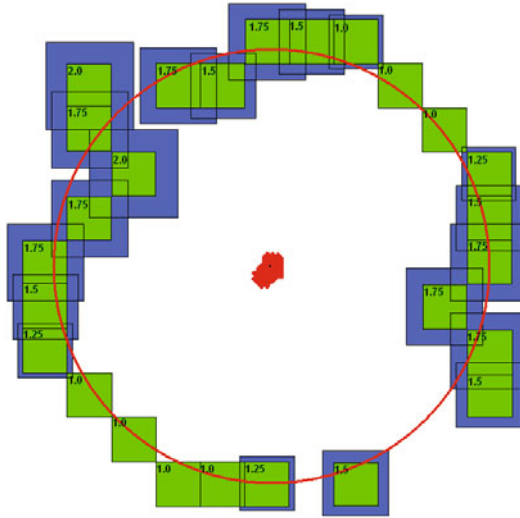


Fig. 4. Circle parameters exhaustive estimation with noisy data

4 Generalized Reflection Symmetry and Biased Rotation Parameter Estimation

As a second application, we show how the SGPB helps to visualize the errors that occur when noisy data are used in the rotation reconstruction algorithm recently proposed by Fontijne et al. [8]. For this we are going to introduce a generalization of the reflection symmetry.

4.1 Bundles and Strips

We define the line bundle \mathbb{F}_i as the set of lines Δ that pass through \mathbb{P}_i with slopes in the interval $[slope(\mathbb{D}_i), slope(\mathbb{D}_{i+1})]$ (or $[slope(\mathbb{D}_{i+1}), slope(\mathbb{D}_{i+2})]$ according to the values of i) as shown in the Figure 2(b). We also define a line strip $strip(\mathbb{P}_i, \mathbb{P}_j)$ by the set of parallel lines bordered by $\Delta_i \in M$ passing through \mathbb{P}_i for $i \in [1; 4]$ and $\Delta_j \in M$ passing through \mathbb{P}_j for $j \in [5; 8]$ as shown in the figure 2(b).

A line in the SGPB can either be characterized as a point inside the dual of the SGPB or by using bundles \mathbb{F}_i and strips. Considering a line D in the SGPB two cases are possible. In the first case, the line D passes through \mathbb{P}_i then D belongs to \mathbb{F}_i . In the second case the line D does not pass through \mathbb{P}_i then there exists \mathbb{P}_j and \mathbb{P}_k such that a translate D' of D passes through \mathbb{P}_j and another translate D'' of D passes through \mathbb{P}_k . In this latter case, D belongs to $strip(\mathbb{P}_j, \mathbb{P}_k)$ (see Figure 2(b)). This leads to a second characterization of a line inside a SGPB:

Property 1 (Lines in the SGPB). *Every line in the SGPB belongs either to a bundle \mathbb{F}_i or is included in a strip strip($\mathbb{P}_k, \mathbb{P}_l$)*

Furthermore, we have the additional property:

Property 2. *The union of the 8 bundles covers the GPB.*

Proof. Let X a point of the GPB. Since the border of the GPB is composed of two convex polylines ∂C_1 and ∂C_2 , there exists two lines Δ_1 and Δ_2 which pass through X and respectively $p_1 \in \partial C_1$ and $p_2 \in \partial C_2$. Obviously, p_1 and $p_2 \in \mathbb{P}_i$. \square

Generalized Reflection Symmetry for a pixel. When considering the perpendicular bisector B of two points p and q , it is obvious that p is the image of q and q the image of p through a reflection symmetry of axis B . Let us define the Generalized Reflection Symmetry $GRS(P_1, P_2)$ of a point as the union of the reflection symmetries of axis the straight lines in the SGPB of two pixels P_1 and P_2 . Not surprisingly, the image of a pixel P_3 by a $GRS(P_1, P_2)$ is not a pixel. Let us characterize the obtained region denoted $\mathcal{C}_{GRS(P_1, P_2)}(P_3)$ (See Figure 6).

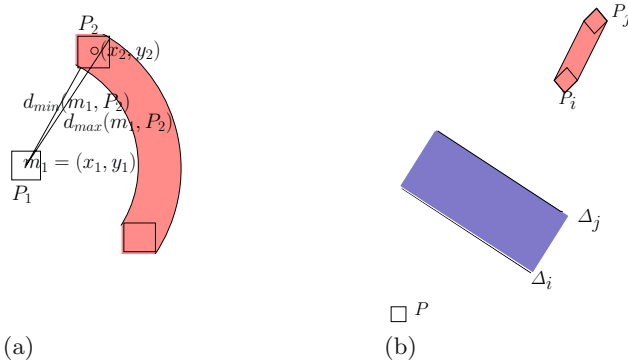


Fig. 5. (a) Image of the pixel P_2 by a partial bundle from m_1 ; (b) Image of the pixel P by the strip(Δ_i, Δ_j)

By construction, a point p of $\mathcal{C}_{GRS(P_1, P_2)}(P_3)$ is the symmetric of a point p_3 belonging to the pixel P_3 with respect to a line Δ that belongs to $SGPB(P_1, P_2)$. More formally, this can be written as:

$$\mathcal{C}_{GRS(P_1, P_2)}(P_3) = \{p \in \mathbb{R}^2 / \exists p_3 \in P_3, \exists \Delta \in SGPB(P_1, P_2), p = sym(p_3, \Delta)\}$$

where $sym(p_3, \Delta)$ denotes the reflection symmetry of the point p_3 with respect to the line Δ . Hence, the following characterization of a point belonging to $\mathcal{C}_{GRS(P_1, P_2)}(P_3)$ is obtained:

Definition 1 (GRS of a pixel). *Let P_1 and P_2 be two pixels centered in (x_1, y_1) and (x_2, y_2) and the associated $SGPB(P_1, P_2)$. Let $\mathcal{C}_{GRS(P_1, P_2)}(P_3)$ be the image of the pixel P_3 by $GRS(P_1, P_2)$. A point (x, y) belongs to $\mathcal{C}_{GRS(P_1, P_2)}(P_3)$ if the following assertions are true:*

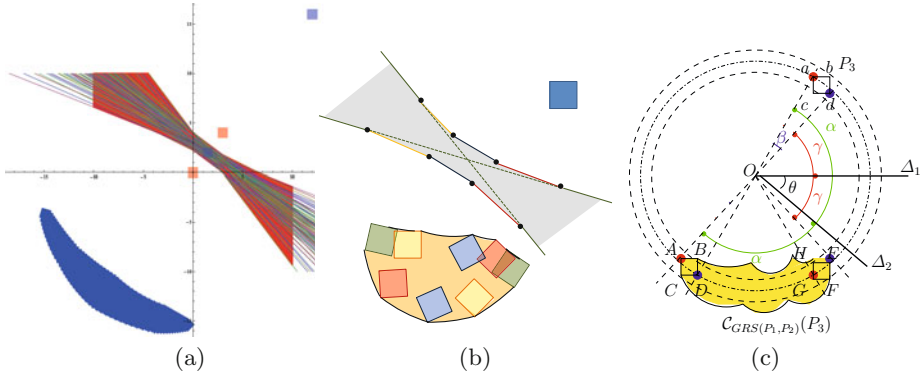


Fig. 6. (a) Generalized Reflection Symmetry of a pixel (12,16) relative to a SGPB between the pixel (0,0) and (3,4); (b) Image of a pixel by a GRS. The obtained shape is composed of ring pieces; (c) Symmetric of a pixel P_3 with respect to the lines Δ_1 and Δ_2 .

- (i) $\exists(x', y')$ such that $(x_3 - 0.5 \leq x' \leq x_3 + 0.5)$ and $(y_3 - 0.5 \leq y' \leq y_3 + 0.5)$
- (ii) (x, y) can be written as $(\frac{-2ac - a^2x' + b^2x' - 2aby'}{a^2 + b^2}, \frac{-2bc - 2abx' + a^2y' - b^2y'}{a^2 + b^2})$ such that $(x_2, \frac{-c - ax_2}{b})$, $(x_1, \frac{-c - ax_1}{b})$, $(\frac{-by_1 - c}{a}, y_1)$, $(\frac{-by_2 - c}{a}, y_2)$ satisfy the distance law (7).

Let us detail a way to construct the GRS of a pixel. The image of a pixel P_3 by a complete¹ line bundle passing through a point p is a ring centered in p defined by the circles centered in p with radii $d_{min}(p, P_3)$ and $d_{max}(p, P_3)$. In our case, the bundle is not complete², the result is then a piece of ring (see Figure 5(a)). Let P_i and P_j be the images of the pixel P by the two lines which define a strip. The image of the pixel P by this line strip is the strip bounded by the images of the pixel relatively to the line passing through the centers of P_i and P_j as shown in the Figure 5(b).

Let P_3 be the pixel of which the image is to be computed. The lines \mathbb{D}_j are the lines bounding the $SGPB(P_1, P_2)$ and \mathbb{P}_i the characteristic points of the $SGPB(P_1, P_2)$. Let I_j be the images of P_3 by \mathbb{D}_j . Since $\mathbb{D}_1 = \mathbb{D}_{10}$ and $\mathbb{D}_5 = \mathbb{D}_6$ (lemma 1), there are only eight different images. Then we consider the images of P_3 by the whole line bundle \mathbb{F}_i that passes through $\mathbb{P}_i = \mathbb{D}_l \cap \mathbb{D}_m$. These images are the ring pieces starting with I_l , ending with I_m of internal radius $d_{min}(\mathbb{P}_i, P_3)$ and external radius $d_{max}(\mathbb{P}_i, P_3)$. We therefore obtain eight ring pieces that form a closed object (since $\mathbb{D}_1 = \mathbb{D}_{10}$ and $\mathbb{D}_5 = \mathbb{D}_6$) as shown in the Figure 6(b).

The last building step consist in integrating the images of P_3 by all the possible strips that are included in the $SGPB(P_1, P_2)$. This leads to a second characterization of the shape $C_{GRS(P_1, P_2)}(P_3)$:

¹ Complete means all the lines are passing through the point.

² That means the slopes of the lines are included between $slope_{min}$ and $slope_{max}$.

Characterization 1 (GRS of a pixel). *A point belongs to the image of a pixel P_3 by the GRS(P_1, P_2) between the pixel P_1 and P_2 if and only if it belongs to the set points defined by the eight pieces of rings centered in \mathbb{P}_i for $1 \leq i \leq 4$ of radius $d_{min}(\mathbb{P}_{i_{1 \leq i \leq 4}}, P_3)$ and centered in \mathbb{P}_i for $5 \leq i \leq 8$ of radius $d_{max}(\mathbb{P}_{i_{5 \leq i \leq 8}}, P_3)$, where the \mathbb{P}_i are the characteristic points of the SGPB.*

The angular arc between the extrema of the region (as defined below) $\mathcal{C}_{GRS(P_1, P_2)}(P_3)$ gives us an upper bound of “noise” that is added to a pixel when its image by a GRS is computed. Let consider the lines Δ_1 and Δ_2 that are the supports of the half-lines \mathbb{D}_1 and \mathbb{D}_5 (see Figure 6(c)). They intersect at the point $O = (\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2})$. On Figure 6(c), the symmetric squares $ABDC$ and $EFGH$ of the square $abcd$ by the lines Δ_1 and Δ_2 are the extrema of the region $\mathcal{C}_{GRS(P_1, P_2)}(P_3)$. As the lines Δ_1 and Δ_2 belong to the bundle of lines that pass through the point O , the squares $ABDC$ and $EFGH$ are contained in the ring centered in O formed by the circles \mathcal{C}_1 and \mathcal{C}_2 of radii $R1 = d_{min}(O, P_3)$ and $R2 = d_{max}(O, P_3)$. Since B and H belong to \mathcal{C}_1 and C, F belong to \mathcal{C}_2 , the points A, D, G and E belong to the circle centered in O of radius $R3 = \frac{R1+R2}{2}$. Then, the angular length between the extrema of $\mathcal{C}_{GRS(P_1, P_2)}(P_3)$ is given by $(\beta + 2\theta)(\frac{R1+R2}{2})$, β and θ respectively denote the angles between the lines Oa and Od and the lines Δ_1 and Δ_2 .

4.2 Rotation Reconstruction Using the SGPB

In this section, the rotation reconstruction algorithm proposed by Fontijne *et al.* [8] is adapted to noisy data using the SGPB. This algorithm is based on the Cartan-Dieudonné theorem [3] that decomposes nD rotations into reflections. Methods based on this approach [2,8] need exact point correspondences and suffer from noisy data.

However, rotation reconstruction methods from point correspondences are used in many application domains such as Computer Vision or body movement analysis. Different approaches can be found in the literature [4,17,9]. Their drawbacks are the impossibility to extend them in dimension higher than two or the subtle geometrical model and computations they need. A simple algorithm based on the decomposition of nD rotations into planar rotations has been developed by some of the authors and it has been shown to be rather robust against noisy data [13,12]. But, contrary to the Fontijne *et al.* [8] algorithm, this approach is not well suited when the data are acquired incrementally.

The purpose of this section is to give some hints on how the SGPB could help to develop an adapted version of the Fontijne *et al.* [8] algorithm to noisy data. For this first attempt, only dimension two is explored but as SGPB is defined for any dimension, extension to nD is expected.

Rotation reconstruction algorithm. To reconstruct the rotation R from n points p_i and their correspondences $p'_i = R(p_i)$ the algorithm proposed in [8] works by finding successive reflections using the Cartan-Dieudonné theorem [3].

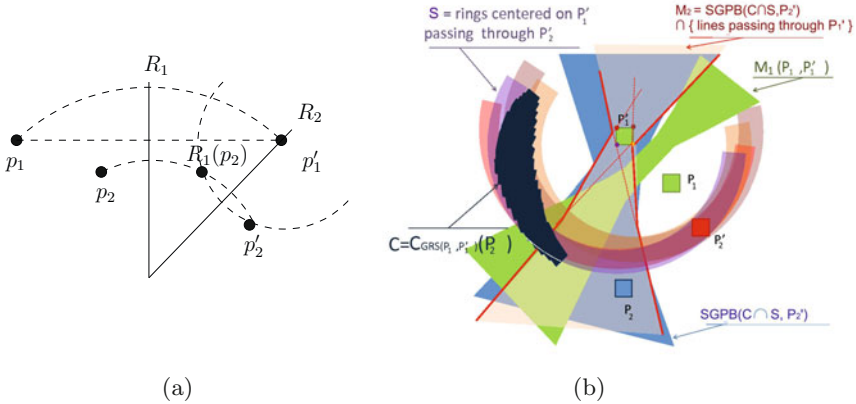


Fig. 7. (a) Reconstructing a 2D-rotation. Note that p'_1 is contained in R_2 , hence $p'_1 = R_2(p'_1) = R_2(R_1(p_1))$. The rotation $R = R_2(R_1(\cdot))$ is a rotation about the intersection of R_1 and R_2 . (b) Reconstruction of a noisy 2D-rotation.

Hence, for each i , we have $p'_i = R(p_i) = R_k(\dots R_2(R_1(p_i)) \dots)$ with $k = \lfloor \frac{n}{2} \rfloor$, where R_i are the reflections³.

The R_i are determined incrementally as perpendicular bisectors. The first reflection R_1 is determined as the perpendicular bisector of p_1 and p'_1 , the second reflection R_2 is the perpendicular bisector of $R_1(p_2)$ and p'_2 .

By construction, the reflections have the property that their composition does not move previously aligned points. For example, as

$$p'_1 = R_1(p_1) = R(p_1) = R_k(\dots R_2(R_1(p_1)) \dots)$$

we must have $R_k(\dots R_2(p'_1) \dots) = p'_1$.

The Figure 7 illustrates the algorithm in dimension two. It must be noted that $p'_1 = R_1(p_1)$ is on the line R_2 and it is the center of circle passing through $R_1(p_2)$ and p'_2 . Now, replacing the perpendicular bisectors with the SGPB could help to adapt the algorithm of Fontijne et al. [8] to noisy data.

Adaptation of the rotation reconstruction algorithm to noisy data. The algorithm described in the previous section supposes that exact point correspondences are provided. However, in the case of noisy data, the reflections became hard to determine. To deal with noisy data, the idea is to replace points by n -dimensional pixels and to introduce the SGPB into the reflection determination process. The expected result is a class of rotations that fit the input data.

Principle of the algorithm (see Figure 7(b)). First the points p_i and their corresponding points p'_i are replaced by n -dimensional pixels P_i and P'_i . Then, the SGPB $M_1 = SGPB(P_1, P'_1)$ (corresponding to R_1 in the exact case) is determined. Using M_1 , the region $C_{GRS}(P_1, P'_1)(P_2)$ is computed. This region corresponds to $R_1(p_2)$ in the exact case.

³ In this section, R_i denotes indistinguishably the reflection and the line that determines the reflection.

In order to restrict $\mathcal{C}_{GRS(P_1, P'_1)}(P_2)$, we use the property that a circle centered on p'_1 passing through p_2 and $R_1(p_2)$ exists (see Figure 7(a)). In the pixel case this means that we have to consider the set S of all circles centered on points of P'_1 passing through P'_2 . So region of interest for the next step is $\mathcal{C}_{GRS(P_1, P'_1)}(P_2) \cap S$.

Now, the $SGBP(\mathcal{C}_{GRS(P_1, P'_1)}(P_2) \cap S, P'_2)$ is computed (see Figure 7(b)). The obtained region must also be restricted with the set L of lines that pass through P'_1 while in the exact case the bisector R_2 passes through p'_1 (see Figure 7(a)).

This finally determines the region $M_2 = SGBP(\mathcal{C}_{GRS(P_1, P'_1)}(P_2) \cap S, P'_2) \cap L$ corresponding R_2 in the exact case. In higher dimension, this process is continued with next point correspondences.

At the end of the algorithm we obtain a class of rotations that are determined by lines chosen in the sequence of the M_i , $1 \leq i \leq n$.

At present, we only have experienced this process in dimension 2 and we face efficiency problems for computing the SGBP between a pixel and a complex region such as the $\mathcal{C}_{GRS(P_1, P'_1)}(P_2)$. Currently, we are working on this efficiency problem and also on the extension of this algorithm to higher dimension.

5 Conclusion

In this paper, we studied the Simplified Generalized Perpendicular Bisector and its properties: whereas the classical Euclidean 2D perpendicular bisector (and all the commonly used medial axis notions), the SGPB of two regions is a 2D surface that collects all the perpendicular bisector of any couple of points of both regions. We have particularly defined the SGPB between two pixels of different sizes. The SGPB is useful to obtain an exhaustive parameter estimation of noisy circles. We have also characterized the lines in the SGPB by showing that the dual of a SGPB is a convex polygon and we have defined a new operation: the Generalized Reflection Symmetry which is a symmetry relatively to a SGPB. This operation allows us to adapt the algorithm of Fontijne et al. [8] to reconstruct rotations while taking noisy data into account. In future works, the GPB will be further studied. There are various notions of discrete bisector that have been proposed. They are, among other applications, used to analyze and filter medial axis [5, 16] where the medial axis of a Jordan curve is in any point equidistant to its borders. These notions of discrete bisectors are discrete curves. How these notions are linked to the Generalized Bisector is one the question we would like to investigate. Especially the links with Voronoï diagrams, medial axis and skeletons seems promising. An extension in higher dimension will also be investigated.

References

1. Andres, E., Largeteau-Skapin, G., Rodríguez, M.: Generalized perpendicular bisector and exhaustive discrete circle recognition (2010) (submitted for publication at Graphical Models)
2. Aragon-Gonzalez, G., Aragon, J.L., Rodriguez-Andrade, M.A., Verde-Star, L.: Reflections, rotations, and pythagorean numbers, vol. 19, pp. 1–14 (2009)

3. Audin, M.: Geometry, pp. 46–49. Springer, Heidelberg (2003)
4. Cheng, P.: Joint rotation between two attitudes in the spherical rotation coordinate system, vol. 37, pp. 1475–1482 (2004)
5. Couprie, M., Coeurjolly, D., Zrou, R.: Discrete bisector function and Euclidean skeleton in 2D and 3D. *Image and Vision Computing* 25(10), 1543–1556 (2007)
6. Dexet, M.: Architecture d'un modeleur discret à base topologique d'objets discret et méthodes de reconstruction en dimensions 2 et 3. Ph.D. thesis, Université de Poitiers (2006)
7. Farouki, R.T., Johnstone, J.K.: Computing point/curve and curve/curve bisectors. In: Fisher, R.B. (ed.) *The Mathematics of Surfaces V*, pp. 327–354. Oxford University, Oxford (1994)
8. Fontijne, D., Dorst, L.: Reconstructing rotations and rigid body motions from points correspondences as a sequence of reflections. In: *Applied Geometric Algebras in Computer Science and Engineering, AGACSE 2010*, Amsterdam, The Netherlands (June 14-16, 2010)
9. Gebken, C., Perwass, C., Sommer, G.: Parameter estimation from uncertain data in geometric algebra. *Advances in Applied Clifford Algebra* 18, 647–664 (2008)
10. Kerautret, B., Lachaud, J.-O.: Multi-scale analysis of discrete contours for unsupervised noise detection. In: Wiederhold, P., Barneva, R.P. (eds.) *IWCIA 2009*. LNCS, vol. 5852, pp. 187–200. Springer, Heidelberg (2009)
11. Peternell, M.: Geometric properties of bisector surfaces. *Graphical Models* 62(3), 202–236 (2000)
12. Richard, A., Fuchs, L., Andres, E., Largeteau-Skapin, G.: Decomposition of n D-rotations: classification, properties and algorithm (2010) (submitted for publication at *Graphical Models*)
13. Richard, A., Fuchs, L., Charneau, S.: An algorithm to decompose n -dimensional rotations into planar rotations. In: Barneva, R., Brimkov, V., Hauptman, H., Natal Jorge, R., Tavares, J. (eds.) *CompIMAGE 2010*. LNCS, vol. 6026, pp. 60–71. Springer, Heidelberg (2010)
14. Rodríguez, M., Largeteau-Skapin, G., Andres, E.: Adaptive pixel resizing for multiscale recognition and reconstruction. In: Wiederhold, P., Barneva, R.P. (eds.) *IWCIA 2009*. LNCS, vol. 5852, pp. 252–265. Springer, Heidelberg (2009)
15. Rodríguez, M., Sere, A., Largeteau-Skapin, G., Andres, E.: Generalized perpendicular bisector and circumcenter. In: Barneva, R., Brimkov, V., Hauptman, H., Natal Jorge, R., Tavares, J. (eds.) *CompIMAGE 2010*. LNCS, vol. 6026, pp. 1–10. Springer, Heidelberg (2010)
16. Talbot, H., Vincent, L.: Euclidean skeletons and conditional bisectors. In: *SPIE*, vol. 1818, pp. 862–876 (1992)
17. Watson, G.: Computing Helmert transformations, vol. 197, pp. 387–394 (2006)

Delaunay Properties of Digital Straight Segments

Tristan Roussillon¹ and Jacques-Olivier Lachaud²

¹ Université de Lyon,
Université Lyon 2, LIRIS, UMR5205, F-69676, France
`tristan.roussillon@liris.cnrs.fr`

² LAMA, UMR CNRS 5127
Université de Savoie, Le Bourget-du-Lac, 73376, France
`jacques-olivier.lachaud@univ-savoie.fr`

Abstract. We present new results concerning the Delaunay triangulation of the set of points of pieces of digital straight lines. More precisely, we show how the triangulation topology follows the arithmetic decomposition of the line slope as well as its combinatorial decomposition (splitting formula). A byproduct is a linear time algorithm for computing the Delaunay triangulation and the Voronoi diagram of such sets.

1 Introduction

Let S be a set of n points in \mathbb{R}^2 . Its *convex hull* is the intersection of every half-planes containing S . A *triangulation* of S is a simplicial decomposition of the convex hull of S where the vertices of the *triangular facets* are elements of S . The *Delaunay triangulation* of S is a triangulation such that each facet satisfies the *Delaunay condition*: the circumcircle of the facet contains no point from S in its interior. Such a triangulation exists for every point set in \mathbb{R}^2 (and more generally in arbitrary dimension), and it is the dual of the Voronoi diagram. It plays a very important role in computational geometry and meshing. A lot of algorithms exist to compute it, and have optimal time complexity $O(n \log n)$ [7].

The Delaunay triangulation of a set of Euclidean points has been deeply studied in computational geometry. The special case of a set of digital points (points in the digital plane \mathbb{Z}^2) is generally not addressed as is, but more as a binary image. In this context, euclidean distance transforms provide a way of computing digital Voronoi diagram *i.e.* each pixel is labeled according to its closest site, but does not give the discrete Voronoi diagram, *i.e.* the set of vertices, edges of the diagram and its topology, and thus do not give the Delaunay triangulation either.

In this paper we study precisely the properties of the Delaunay triangulation of specific subsets of the digital plane \mathbb{Z}^2 , which are digital equivalents of straight lines and segments. The *standard line* $D(a, b, \mu)$ of slope $\frac{a}{b}$ and intercept μ is the set of point $(x, y) \in \mathbb{Z}^2$ verifying $\mu \leq ax - by < \mu + |a| + |b|$ with a, b, μ integer and $\gcd(a, b) = 1$ [9]. The points verifying $ax - by = \mu$ (resp.

$ax - by = \mu + |a| + |b| - 1$) are called the *upper* (resp. *lower*) *leaning points*. The connected part of a standard line $D(a, b, \mu)$ between two consecutive upper leaning points U and $U' = U + (b, a)$ is called a *pattern*. It is a 4-connected path joining U and U' , conveniently denoted by UU' . Its *staircase representation* is the polygonal line joining the points of the pattern in the same order as the 4-connected path does. The chain code of a pattern is a *Christoffel word*. The *slope* of a pattern is the slope of its vector $\overrightarrow{UU'} = (b, a)$, i.e. $\frac{a}{b}$. These standard definitions are illustrated on Fig. 1a. Note that any digital straight segment has a unique decomposition into patterns, which makes patterns very important for analyzing digital shapes.

We show in Section 2 that the Delaunay triangulation of a pattern has very specific properties. They are related to the specific geometry of points within a pattern. From an equivalent arithmetic point of view, they are related to the continued fraction of the pattern slope or, from a combinatorial point of view, to the splitting formula (e.g. see [5]). With these properties, we deduce in Section 3 a linear time algorithm to compute the Delaunay triangulation and the Voronoi diagram of such sets of points. The presented results may have several interesting applications in computational geometry and digital shape analysis which are discussed in Section 4.

2 Delaunay Properties of Patterns

Before stating precisely the main theorem, we introduce briefly some further notions. From now on, we restrict our study to the first quadrant. Let UU' be the pattern of the standard line $D(a, b, 0)$ between $U = (0, 0)$ and $U' = (b, a)$ with $a > 0$ and $b > 0$.

2.1 Triangulation of a Pattern

Let us denote by $\mathcal{H}(UU')$ the convex hull of UU' . The staircase representation of UU' is by definition below the straight segment $[UU']$. Therefore, $\mathcal{H}(UU')$ can be divided into two parts lying on both sides of its staircase representation: the upper part $\mathcal{H}^+(UU')$, whose boundary contains $[UU']$, in light gray fig 1b, and the lower one $\mathcal{H}^-(UU')$, in dark gray fig 1b.

Let us denote by $\mathcal{T}(UU')$ the Delaunay triangulation of UU' . We will see that it can similarly be divided into two parts $\mathcal{T}^+(UU')$ and $\mathcal{T}^-(UU')$, separated by the staircase representation. Fig. 1b shows the Delaunay triangulation of a pattern of slope $\frac{4}{7}$.

We introduce here three equivalent definitions of the main facet of a pattern (see Fig. 2).

Definition 1 (main facet). *The main facet of a pattern UU' is the triangle UBU' , where B has the following equivalent definitions:*

(geometric) *the digital point B within $UU' \setminus \{U, U'\}$ that has the shortest orthogonal distance to the segment $[UU']$;*

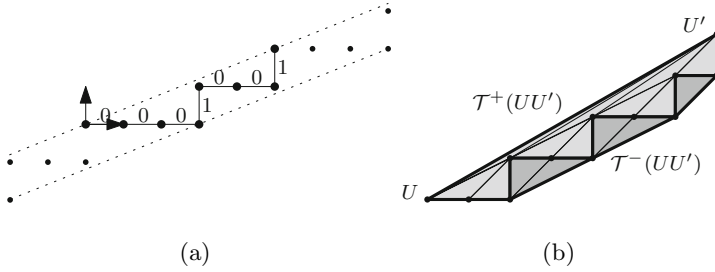


Fig. 1. (a) The points of $D(2, 5, 0)$ are depicted with black disks. The two dotted straight lines pass through the upper and lower leaning points. A pattern is highlighted with bigger disks. The letters of its chain code are indicated near its staircase representation, which is depicted with a solid line. (b) The Delaunay triangulation of the pattern UU' is divided into the upper part $\mathcal{T}^+(UU')$, which contains the facets lying in $\mathcal{H}^+(UU')$, in light gray, and the lower part $\mathcal{T}^-(UU')$, which contains the facets lying in $\mathcal{H}^-(UU')$, in dark gray.

- (arithmetic) the lower Bezout point B to the vector $\overrightarrow{UU'}$;
- (combinatorial) the separation point B of the splitting formula (see [10], chap. 4).

These characterizations are shown equivalent later in the paper.

Definition 2 (facets of a pattern). *The facets of the pattern UU' , denoted by $\mathcal{F}(UU')$, is defined recursively as the union of the main facet UBU' of UU' and the facets of the pattern UB if $a > 1$ and the facets of the pattern BU' if $b > 1$.*

Note that this recursive definition is consistent because (i) UB and BU' are both patterns (see Proposition 1 later), (ii) $a + b$ strictly decreases so that both a and b reach 1 and the recursion terminates.

2.2 Main Result

We can now state precisely our main result.

Theorem 1. *The facets of the pattern UU' is a triangulation of $\mathcal{H}^+(UU')$. Furthermore, each facet satisfies the Delaunay property, i.e. the circumcircle of each triangular facet of $\mathcal{F}(UU')$ contains none of the points of the pattern UU' in its interior.*

In other words, the facets of the pattern UU' is exactly $\mathcal{T}^+(UU')$.

We prove this theorem in the following subsections. We prove first that the facets form a triangulation (Theorem 2), then that each of them has the Delaunay property (Theorem 3).

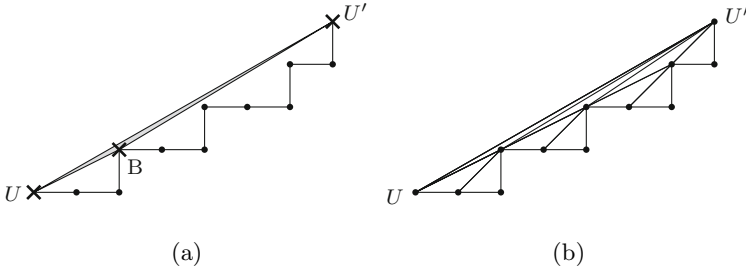


Fig. 2. (a) The gray area, bounded by the triangle UBU' , is the main facet of UU' . Note that $B = b^-(UU')$ belongs to UU' and that UB and BU' are both patterns. (b) The facets of UU' are defined by induction over the patterns included in UU' .

2.3 The Facets of a Pattern Form a Triangulation

We recall that the *remainder* with respect to the standard line of slope $\frac{a}{b}$ is a function $r_{a,b} : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ such that $r_{a,b}(x, y) = ax - by$. In the sequel, $r_{a,b}(\cdot)$ is simplified into $r(\cdot)$ when the remainder refers to the standard line $D(a, b, 0)$ containing UU' .

Definition 3 (support of a pattern). *The support of the pattern UU' is the set of lattice points $S(UU')$ lying in a strip bounded by two straight lines orthogonal to the straight line (UU') and passing through U and U' , i.e. $S(UU') = \{(x, y) \in \mathbb{Z}^2 \mid 0 \leq bx + ay \leq a^2 + b^2 - 1\}$.*

Given an integer k , let the straight line \mathcal{L}_k goes through the lattice points X of remainder $r(X) = k$. The lattice points of \mathcal{L}_k are regularly spaced of a distance equal to $\sqrt{a^2 + b^2}$. Since the points of $S(UU')$ lie in a strip of width strictly less than $\sqrt{a^2 + b^2}$, \mathcal{L}_k contains only one point of $S(UU')$ for each k . Thus, we may define (see Fig. 2a):

Definition 4 (Bezout point). *The lower Bezout point of the pattern UU' , denoted by $b^-(UU')$ is the unique point of $S(UU')$ whose remainder equals to 1.*

Note that $b^-(UU')$, whose remainder equals to 1, belongs to the pattern UU' contained in $D(a, b, 0)$ if and only if $|a| + |b| \geq 2$, i.e. $a \neq 0$ and $b \neq 0$. Considering that the orthogonal distance of a lattice point P with respect to the line UU' is $|r(P)/\sqrt{a^2 + b^2}|$, it is clear that the Bezout point to UU' is the point of $UU' \setminus \{U, U'\}$ closest to the segment $[UU']$, which proves the equivalence between the geometric and arithmetic definition of the main facet (Definition 1).

Note that the main facet of a pattern never contains any lattice points:

Proposition 1. *Let X be a lattice point defined with a positive integer k such that $r(X) = k$. The triangle UXU' contains in its interior one or more lattice points if and only if $k > 1$.*

Proposition 1, which is directly proved with Pick formula, explains why, $b^-(UU')$ belongs to UU' , the 4-path joining U to $b^-(UU')$ as well as the one

joining $b^-(UU')$ to U' , are both patterns. Therefore the inductive definition of the facets of a pattern (Definition 2) makes sens. The number of facets of a pattern is finite because $\mathcal{F}(UU')$ coincides with $\mathcal{H}^+(UU')$ (compare Fig. 2b and Fig. 1b), as shown below:

Theorem 2. *The facets of a pattern UU' is a triangulation of $\mathcal{H}^+(UU')$.*

Proof. We shall prove this assertion by structural induction on the recursive definition of the facets of a pattern.

(i) *Base case:* Let us take a pattern with $a = 1$ and $b = 1$. The facets of UU' are then reduced to the main facet UBU' which is the triangle $(0, 0), (0, 1), (1, 1)$. It is equal to its convex hull $\mathcal{H}^+(UU')$ and is thus its triangulation.

(ii) *Inductive step:* Given a pattern UU' with $a > 1$ or $b > 1$, let us assume that the facets of UB is a triangulation of $\mathcal{H}^+(UB)$ and the facets of UB' is a triangulation of $\mathcal{H}^+(BU')$. We shall prove that the facets of UU' is a triangulation of $\mathcal{H}^+(UU')$.

First of all, since $\mathcal{H}^+(UB)$ is below $[UB]$ and $\mathcal{H}^+(BU')$ is below $[BU']$, both have thus an empty intersection with the interior of the triangle UBU' (the main facet of UU'). Furthermore, $\mathcal{H}^+(UU')$ forms a simple polygon (Euler characteristics is 2) since it is the union of three simple polygons, with one more edge ($[UU']$) and one more face (UBU') than the union of $\mathcal{H}^+(UB)$ and $\mathcal{H}^+(BU')$. Lastly, the boundary edges of the facets of UU' are exactly the boundary edges of $\mathcal{H}^+(UU')$: the main facet has its edge $[UU']$ that is the upper edge of $\mathcal{H}^+(UU')$, while the lower edges of $\mathcal{H}^+(UB)$ and $\mathcal{H}^+(BU')$ define the staircase representation of UU' , which is by definition the lower edges of $\mathcal{H}^+(UU')$.

The facets of UU' thus define a triangulation of a simple polygon whose (only) boundary is the same as the boundary of $\mathcal{H}^+(UU')$. It is thus a triangulation of $\mathcal{H}^+(UU')$. □

2.4 Delaunay Condition for Each Facet

It remains to prove that each facet of $\mathcal{F}(UU')$ satisfies the Delaunay condition. In the whole subsection, we denote by B the lower Bezout point of the pattern UU' ($a > 0$ and $b > 0$). Let us begin by a lemma that completes Proposition 1:

Lemma 1. *For all $P \in S(UU')$ with $r(P) > 1$, the triangle UPU' contains B .*

Lemma 1 and its proof are illustrated in Fig. 3a.

Proof. For all $P \in S(UU')$ such that $r(P) > 1$, we prove below that the slope of \overrightarrow{UP} is smaller than the one of \overrightarrow{UB} and similarly that the slope of $\overrightarrow{PU'}$ is greater than the one of $\overrightarrow{BU'}$, which prove that the triangle UPU' contains the triangle UBU' and thus B .

Given a positive integer k , let $B_k = k\overrightarrow{UB}$. We have $B = B_1 \in S(UU')$ and $r(B_k) = k$. Let k_0 be the greatest $k > 1$ such that $B_{k-1} \in S(UU')$ ($k_0 = 3$ in Fig. 3a). The lattice point B_{k_0} is not in $S(UU')$ and $k_0 \geq 2$. Hereafter, we independently deal with the points of $S(UU')$ of remainder ranging from 2 to $k_0 - 1$ (i) and greater than or equal to k_0 (ii).

(i) For the $k_0 - 2$ points P of $S(UU')$ such that $1 < r(P) < k_0$, \overrightarrow{UP} has the same slope as \overrightarrow{UB} .

(ii) For any point X , the tangent of the angle $\angle XU'U'$ is equal to the ratio between the determinant of $\overrightarrow{UX}, \overrightarrow{UU'}$ and the scalar product of $\overrightarrow{UX}, \overrightarrow{UU'}$.

The scalar product with $\overrightarrow{UU'}$ defines the length of the orthogonal projection on the line (UU') . It is obvious that the length of the projection of a point outside the orthogonal strip $S(UU')$ is greater than the length of the projection of any point P within this strip, thus $\overrightarrow{UB_{k_0}} \cdot \overrightarrow{UU'} > \overrightarrow{UP} \cdot \overrightarrow{UU'}$.

On the other hand, the determinant between $\overrightarrow{UB_{k_0}}$ and $\overrightarrow{UU'}$, which is equal to $r(B_{k_0}) = k_0$, is smaller than or equal to the determinant between \overrightarrow{UP} and $\overrightarrow{UU'}$ for all $P \in S(UU')$ such that $r(P) \geq k_0$.

$$\tan(\angle BU'U') = \tan(\angle B_{k_0}UU') = \frac{\det(\overrightarrow{UB_{k_0}}, \overrightarrow{UU'})}{\overrightarrow{UB_{k_0}} \cdot \overrightarrow{UU'}} < \frac{\det(\overrightarrow{UP}, \overrightarrow{UU'})}{\overrightarrow{UP} \cdot \overrightarrow{UU'}} = \tan(\angle PU'U').$$

As a consequence, the slope of \overrightarrow{UP} is smaller than the one of \overrightarrow{UB} for all $P \in S(UU')$ such that $r(P) > 1$. We can similarly show that the slope of $\overrightarrow{PU'}$ is greater than the one of $\overrightarrow{BU'}$, which concludes the proof. □

Lemma 1 is used in Lemma 2 and Lemma 4.

Lemma 2. *Let \mathcal{D} be a disk whose boundary passes through U and U' and whose center is located above (UU') . Let $\partial\mathcal{D}$ be its boundary. $\mathcal{D} \setminus \partial\mathcal{D}$ contains a lattice point below or on (UU') if and only if it contains (at least) B (Fig. 3.b).*

Proof. Given a positive integer k , let \mathcal{L}_k be the straight line passing through the points of remainder k . Let $S_{\mathcal{D}}$ be the set of lattice points of strictly positive remainder contained in \mathcal{D} . Since the center of \mathcal{D} is located above (UU') , the length of the segment $\mathcal{L}_k \cap \mathcal{D}$ is smaller than $\sqrt{a^2 + b^2}$ for all $k > 0$. Thus $S_{\mathcal{D}} \subset S(UU')$ (Fig. 3.b).

Due to Lemma 1 (Fig. 3.a), $\angle UPU' \leq \angle UBU'$ for all $P \in S(UU')$ such that $r(P) \geq 1$, which concludes for all lattice points strictly below (UU') .

Since $[UU']$ does not contain any lattice points, except U and U' , $\mathcal{D} \setminus \partial\mathcal{D}$ does not contain any lattice point on (UU') , which concludes. □

Let us now introduce the following key definition:

Definition 5 (background of a pattern). *The background of the pattern UU' is the set of lattice points located below or on the straight line (UB) , or below or on the straight line (BU') .*

Definition 5, illustrated in Fig. 4, is used in Lemma 3 and Lemma 4.

Lemma 3. *Let \mathcal{D} be a disk whose boundary $\partial\mathcal{D}$ is the circumcircle of UBU' . The disk \mathcal{D} contains none of the background points of UU' , except U, U', B , which are all located on $\partial\mathcal{D}$ (Fig. 4.a).*

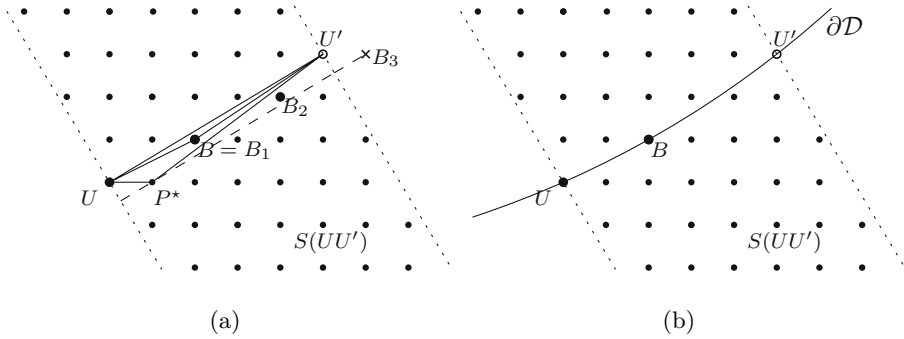


Fig. 3. (a) Illustration of Lemma 1. The triangle UP^*U' contains B because P^* belongs to $S(UU')$ (depicted by the set of black disks) and $r(P^*) > 1$. The points $B_k = k\overrightarrow{UB}$ (indicated for $k = 1, 2, 3$) are used in the proof of Lemma 1 to show that the slope of \overrightarrow{UP} is smaller than the one of \overrightarrow{UB} for all $P \in S(UU')$ such that $r(P) > 1$. (b) Illustration of Lemma 2. \mathcal{D} is a disk whose boundary $\partial\mathcal{D}$ pass through U, U' and B . The interior $\mathcal{D} \setminus \partial\mathcal{D}$ does not contain any lattice points located below or on the straight line (UU') due to Lemma 1.

Proof. Let B_l (resp. B_r) be the lower Bezout point of UB (resp. BU'). The proof is based on the two following arguments:

1. \mathcal{D} contains neither B_l nor B_r .
2. \mathcal{D} contains neither the set of lattice points located below or on (UB) , nor the set of lattice points located below or on (BU') .

Assuming 1) is true, argument 2) follows from Lemma 2 applied once for pattern UB and once for pattern BU' , which in turn concludes the proof due to the definition of background (Definition 5).

Let us now prove argument 1). Moreover let us focus on B_l because the proof about B_r is similar. According again to Lemma 2, it is enough to show that the remainder of B_l is strictly positive so as to show it is below (UU') .

If $a = 1, B = (1, 0), B_l = (0, -1)$ and $r(B_l) = b$. Thus, $r(B_l) \geq 1$.

If $a > 1, B_l$ belongs to the pattern UB , which belongs to the pattern UU' . By definition, all the points $X \in UU' \setminus U'$ belong to $S(UU')$ and have a remainder $r(X)$ ranging from 0 to $|a| + |b| - 1 > 1$. Moreover, given an integer k , there is only one point $X \in S(UU')$ such that $r(X) = k$. Since $B_l \neq B$, we conclude that $r(B_l) > 1$. □

Lemma 4. *The background of UU' is contained in the background of UB if $a > 1$ and in the one of BU' if $b > 1$ (Fig. 4.b).*

Proof. Let us assume that $a > 1$, the case where $b > 1$ being symmetric.

Let B_l be the lower Bezout point of UB . As in the proof of Lemma 3, it can be shown that $r(B_l) > 1$. Due to Lemma 1, the triangle UB_lU' contains the triangle UBU' .

As a consequence, the slope of $\overrightarrow{B_l B}$ is greater than the one of $\overrightarrow{BU'}$. Since the triangle $UB_l B$ does not contain any lattice points due to Proposition 1, the set of lattice points whose x-coordinate is greater than B_l and lying below or on the straight line $(B_l B)$ contains the set of lattice points whose x-coordinate is greater than B_l and lying below or on the straight line (BU') .

Similarly the slope of $\overrightarrow{UB_l}$ is smaller than the one of \overrightarrow{UB} . Since the triangle $UB_l B$ does not contain any lattice points due to Proposition 1, the set of lattice points whose x-coordinate is smaller than B_l and lying below or on the straight line (UB_l) contains the set of lattice points whose x-coordinate is smaller than B_l and lying below or on the straight line (UB) .

Due to the definition of background (Definition 5), we can conclude that the background of UB contains the background of UU' (like in Fig. 4b), which concludes the proof. □

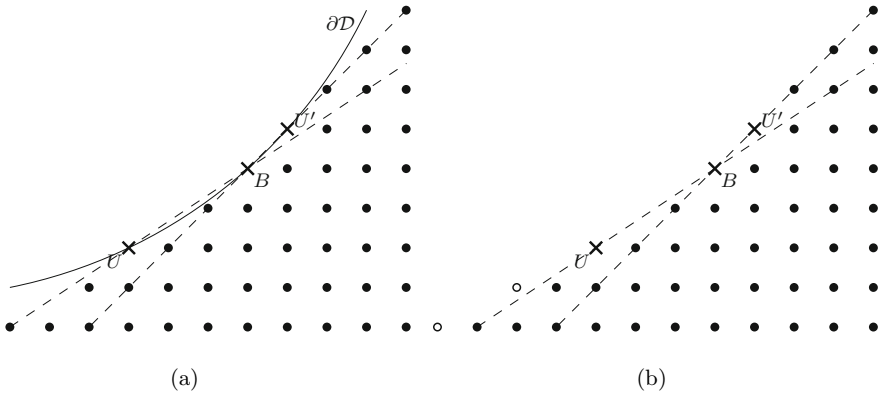


Fig. 4. (a) Illustration of Lemma 3. The set of lattice points located below UB or BU' , depicted with black disks, is the background of UU' . The disk \mathcal{D} passing through U , B , U' contains none of the points of the background UU' . (b) Illustration of Lemma 4. The background of UU' is contained in the background of UB , which has extra points, the two visible ones are depicted with white disks.

We can now prove the Delaunay condition for each facet of $\mathcal{F}(UU')$:

Theorem 3. *The circumcircle of each triangular facet of $\mathcal{F}(UU')$ contains none of the background points of UU' in its interior.*

Proof. We shall prove this assertion by structural induction on the recursive definition of the facets of a pattern.

(i) *Base case:* Let us take a pattern with $a = 1$ and $b = 1$. The facets of UU' are then reduced to the main facet UBU' , which is the triangle $(0, 0), (0, 1), (1, 1)$. The property is trivial by definition of the circumcircle.

(ii) *Inductive step:* Let UB and BU' be two patterns such that B is the lower Bezout point of the pattern UU' with $a > 1$ or $b > 1$. Let us assume that Theorem 3 is true for UB and BU' . We shall prove that Theorem 3 is true for UU' .

Lemma 3 concludes for the main facet UBU' . We observe now the facets of UU' that are also facets of UB or BU' . Since $a > 1$ or $b > 1$, the background of UB or BU' contains the one of UU' due to Lemma 4. Since the circumcircle of each triangular facet of $\mathcal{F}(UB)$ (resp. $\mathcal{F}(BU')$) contains none of the background points of UB (resp. BU') in its interior due to the induction hypothesis, it does not contain any background point of UU' either. We can thus conclude. \square

Since UU' is contained in its background (UBU' does not contain any lattice point due to Proposition 1), Theorems 2 and 3 clearly imply Theorem 1. The facets of the pattern UU' coincides with the upper part of the Delaunay triangulation of UU' , i.e. $\mathcal{F}(UU') = \mathcal{T}^+(UU')$. It remains only to show that the combinatorial characterization of the main facet (Definition 1) is the same as either the arithmetic one or geometric one. This is done in the next section.

3 Applications

In this section, we explain how to efficiently compute the Delaunay triangulation or the Voronoi diagram of the pattern UU' from the continued fraction expansion of its slope.

3.1 Continued Fraction

Let $[q_0; \dots, q_i, \dots, q_n]$ (with $q_n > 1$) be the quotients and $(b_0, a_0), \dots, (b_i, a_i), \dots, (b_n, a_n)$ be the convergent vectors of the continued fraction expansion of $\frac{a}{b}$. Note that (b_n, a_n) is the closest approximation of (b, a) (without being equal to (b, a)), because q_n is assumed to be greater than 1. Let (b_{n+1}, a_{n+1}) be equal by convention to (b, a) . The sequence of convergents verifies:

$$\forall i \in 1, \dots, n, (b_{i+1}, a_{i+1}) = (b_{i-1}, a_{i-1}) + q_i(b_i, a_i) \tag{1}$$

Moreover, for all $i \in 1, \dots, n$, the determinant of (b_i, a_i) and (b_{i+1}, a_{i+1}) is equal to 1 if i is odd and -1 if i is even.

For instance, the continued fraction representation of $\frac{3}{5}$ is $[0; 1, 1, 2]$ and the sequence of its convergent vectors is $(0, 1), (1, 0), (1, 1), (2, 1)$. Note that $(5, 3) = (1, 1) + 2(2, 1)$ (and $(2, 1) = (1, 0) + 1(1, 1)$, etc.) as expected from eq. 1. In addition, $5 \cdot 1 - 3 \cdot 2 = -1$ (and $2 \cdot 1 - 1 \cdot 1 = 1$, etc.). This suggests that the lower Bezout point B is available from the quotients and the convergent vectors of the continued fraction expansion of the slope $\frac{a}{b}$ of UU' and indeed:

$$\overrightarrow{UU'} = \overrightarrow{UB} + \overrightarrow{BU'} = (b_n, a_n) + ((b_{n-1}, a_{n-1}) + (q_n - 1)(b_n, a_n)) \tag{2}$$

Eq. 2 is a formulation in terms of convergent vectors of the *splitting formula*, originally expressed in terms of quotients [10, Theorem 4.1]. The combinatorial characterization of the main facet of Definition 1 is therefore the same as the arithmetic one.

Note that $(b_n, a_n) = \overrightarrow{UB}$ if n is odd, but $(b_n, a_n) = \overrightarrow{BU'}$ if n is even. For instance, if $\overrightarrow{UU'} = (5, 3)$, the last convergent vector is $\overrightarrow{UB} = (2, 1)$ and $\overrightarrow{BU'} = (1, 1) + (2 - 1) \cdot (2, 1) = (3, 2)$.

Furthermore, the description of the slopes of UB and BU' can be easily deduced from the one of UU' : the quotients and convergent vectors associated to the slope of UB is $[q_0; \dots, q_{n-1}]$ and $(b_0, a_0), \dots, (b_{n-1}, a_{n-1})$ if n is odd, $[q_0; \dots, q_n - 1]$ and $(b_0, a_0), \dots, (b_n, a_n)$ if n is even, whereas the quotients and convergent vectors associated to the slope of BU' is $[q_0; \dots, q_n - 1]$ and $(b_0, a_0), \dots, (b_n, a_n)$ if n is odd, $[q_0; \dots, q_{n-1}]$ and $(b_0, a_0), \dots, (b_{n-1}, a_{n-1})$ if n is even.

3.2 Computation of the Delaunay Triangulation

The computation of the Delaunay triangulation of $\mathcal{T}^-(UU') = \mathcal{F}(UU')$ is made by a simple recursive algorithm, which may be coarsely describe as follows: find $B = b^-(UU')$, the lower Bezout point of UU' , using the extended Euclidean algorithm, add the triangular facet UBU' to $\mathcal{F}(UU')$ and split UU' into UB and BU' in order to recursively apply the procedure on sub-patterns. The algorithm stops when the sub-patterns are as small as two consecutive points of UU' .

Though, it is enough to run only once the extended Euclidean algorithm because the sequence of convergent vectors of any sub-patterns is computed in constant time from its parent pattern as said in the previous subsection.

To sum up, the extended Euclidean algorithm is only run once at the beginning in $\mathcal{O}(\log^2(\max(|a|, |b|)))$. Then, each step requires $\mathcal{O}(1)$ time to draw a new triangular facet. Since there are exactly $|a| + |b| + 1$ points and $|a| + |b| - 1$ facets, the whole algorithm is in $\mathcal{O}(|a| + |b|)$, whereas any classic algorithm coming from computational geometry is in $\mathcal{O}((|a| + |b|) \log(|a| + |b|))$ in the RAM model.

To end, note that $\mathcal{T}^-(UU')$, the lower part of the Delaunay triangulation of UU' , is the union of the facets of all the reversed patterns LL' such that the straight segment $[LL']$ is an edge of the lower convex hull of UU' . The slopes of such edges are also given by the continued fraction expansion of the slope of UU' . The connection between continued fractions and the convex hull of the lattice points located above or below a straight segment was already noticed by Klein in 1895 [3].

We do not provide further details due to lack of space, but the algorithm has been implemented in Lua and is available on the web¹. It may be run as an ipelet in Ipe in order to draw the delaunay triangulation of any pattern described by a straight segment.

3.3 Computation of the Voronoi Diagram

The partial Voronoi diagram (Fig. 5.a), which is dual of $\mathcal{T}^+(UU') = \mathcal{F}(UU')$, is also computed by a simple recursive algorithm, because each vertex of the

¹ <http://liris.cnrs.fr/tristan.roussillon/code/delaunay.lua>

diagram is the center of the circumcircle of a triangular facet of $\mathcal{F}(UU')$ and each edge of the diagram belongs to the bisector of an edge of a triangular facet of $\mathcal{F}(UU')$.

The algorithm may be coarsely described as follows: start with the point at infinity along the bisector of $[UU']$, then find $B = b^-(UU')$, the lower Bezout point of UU' , using the extended Euclidean algorithm, link Ω , the center of the circumcircle of UBU' , to the previous vertex and split UU' into UB and BU' in order to recursively apply this procedure on sub-patterns.

Let $B = (\beta, \alpha)$. Simple calculations lead to:

$$\Omega = \left(\frac{a \cdot (\alpha^2 + \beta^2) - (a^2 + b^2) \cdot \alpha}{2}, \frac{(a^2 + b^2) \cdot \beta - b \cdot (\alpha^2 + \beta^2)}{2} \right)$$

For instance, if $U' = (5, 3)$ and $B = (2, 1)$, $\Omega = (\frac{-19}{2}, \frac{43}{2})$ (Ω is the leftmost vertex in Fig. 5.a).

To sum up, the partial Voronoi diagram dual of $\mathcal{T}^+(UU')$ is a planar binary tree whose vertices have integer or half-integer coordinates and whose edges have a slope $-\frac{b_k}{a_k}$ such that $\frac{a_k}{b_k}$ is a fraction lying on the path going from the Stern-Brocot tree root to the fraction $\frac{a}{b}$ ($\frac{3}{5}$ in Fig. 5.b).

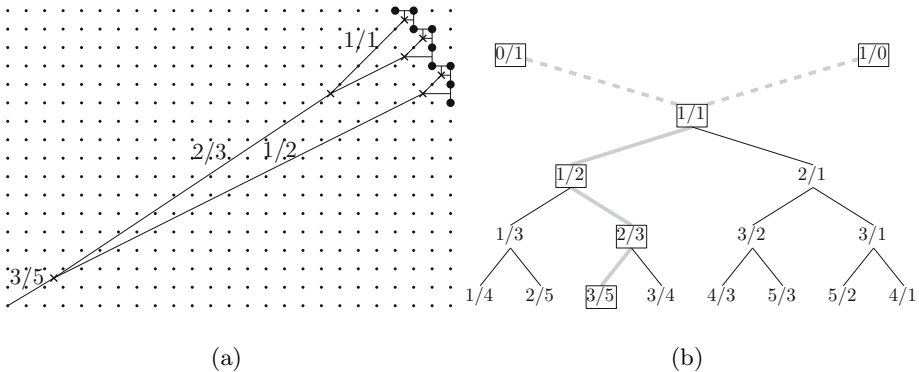


Fig. 5. The partial Voronoi diagram of a pattern of slope $\frac{3}{5}$ is depicted in (a). The figure is turned 90° to the left so that the slopes of the edges are those lying on the path highlighted in (b), which goes from the Stern-Brocot tree root to the fraction $\frac{3}{5}$.

4 Discussion

The presented results may have several interesting applications in digital shape analysis. First of all, it may lead to new algorithms for computing the Delaunay triangulation of a digital shape, starting from a decomposition of its boundary into digital straight segments, for instance the one given by the minimum length polygon (see [8] for a recent algorithm). The input data would then be some $\Theta(n^{\frac{2}{3}})$ instead of n for digitization of smooth shapes. The Delaunay triangulation contains all the α -shapes of its vertices [4], which is of great interest for

shape reconstruction. The present work is also an essential step for enumerating the maximal digital circular arcs along digitizations of Euclidean shapes, since the center of these arcs are related to the Voronoi vertices. This enumeration would help for determining the multigrid convergence of several discrete curvature estimators, similarly to the number of maximal segments that was the key ingredients to determine the multigrid convergence of tangent estimators. Our results may also help in determining properties of the λ -medial axis [2] and its digital counterpart [1]. Finally, a new class of geometric estimators from a set of points relies uniquely on the local shape of the Voronoi diagram [6]. Our results clearly avoid global Voronoi computation and may even give partial analytic quantities for that kind of applications.

References

1. Chaussard, J., Couprie, M., Talbot, H.: A discrete λ -medial axis. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 421–433. Springer, Heidelberg (2009)
2. Chazal, F., Lieutier, A.: The λ -medial axis. *Graph. Models* 67(4), 304–331 (2005)
3. Davenport, H.: *The Higher Arithmetic*. Cambridge University Press, Cambridge (1992)
4. Edelsbrunner, H., Kirkpatrick, D.G., Seidel, R.: On the Shape of a Set of Points in the Plane. *IEEE Trans. on IT* 29(4), 551–559 (1983)
5. Klette, R., Rosenfeld, A.: Digital straightness – a review. *Discrete Applied Mathematics* 139(1-3), 197–230 (2004)
6. Mérigot, Q., Ovsjanikov, M., Guibas, L.: Robust voronoi-based curvature and feature estimation. In: *Proc. SIAM/ACM Joint Conference on Geometric and Physical Modeling (SPM 2009)*, pp. 1–12. ACM, New York (2009)
7. Preparata, F.P., Shamos, M.I.: *Computational geometry: an introduction*. Springer, Heidelberg (1985)
8. Provençal, X., Lachaud, J.O.: Two linear-time algorithms for computing the minimum length polygon of a digital contour. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 104–117. Springer, Heidelberg (2009)
9. Reveillès, J.P.: *Géométrie discrète, calcul en nombres entiers et algorithmique*. Thèse d'état, Université Louis Pasteur, Strasbourg, France (1991) (in french)
10. Voss, K.: *Discrete Images, Objects, and Functions in \mathbb{Z}^n* . Springer, Heidelberg (1993)

Computing the Characteristics of a SubSegment of a Digital Straight Line in Logarithmic Time

Mouhammad Said^{1,2} and Jacques-Olivier Lachaud¹

¹ Laboratoire de Mathématiques, UMR 5127 CNRS, Université de Savoie,
73376 Le Bourget du Lac, France

{mouhammad.said, jacques-olivier.lachaud}@univ-savoie.fr

² LAIC, Univ. Clermont-Ferrand,
IUT, Campus des Cézeaux, 63172 Aubière Cedex, France

Abstract. We address the problem of computing the exact characteristics of any subsegment of a digital straight line with known characteristics. We present a new algorithm that solves this problem, whose correctness is proved. Its principle is to climb the Stern-Brocot tree of fraction in a bottom-up way. Its worst-time complexity is proportionnal to the difference of depth of the slope of the input line and the slope of the output segment. It is thus logarithmic in the coefficients of the input slope. We have tested the effectiveness of this algorithm by computing a multiscale representation of a digital shape, based only on a digital straight segment decomposition of its boundary.

Keywords: standard lines, digital straight segment recognition, Stern-Brocot tree.

1 Introduction

Digital Straight Lines (DSL) and Digital Straight Segments (DSS) are useful to describe the geometry of a digital shape (coding, discrete geometric estimators, feature detection) and this explains why they have been so deeply studied (see the survey [9] or [8]). When a straight line is digitized on a grid, we obtain a sequence of grid points defining a digital straight line segment. Methods of recognizing digital straight segments are known since long. In one of the first methods, Freeman [7] suggested to analyze the regularity in the pattern of the directions in the chain code [6] of a digital curve. Anderson and Kim [1] have presented a deep analysis of the properties of DSS and suggested a different algorithm based on calculating the convex hull of the points of digital curves to be analyzed. In [10], Kovalevsky presented a *new classification of digital curves* into boundary curves and visual curves. Boundary curves and lines are a useful mean for fast drawing of regions defined by their boundaries. Modern DSS recognition algorithms achieves a computational complexity of $O(n)$, if n is the number of input points and with a computing model where arithmetic operations takes $O(1)$ time. Interestingly, this complexity is obtained by algorithms based on arithmetic properties [12,11], combinatorial properties [16], or dual-space construction [5]. These algorithms

are optimal, when no further information is known. However, in some case, we may already know that a set of points is included in some DSL of known characteristics. This happens for instance when computing the multiresolution of a digital object, since analytic formulas give the multiresolution of DSL. We assume that the digital shape was previously polygonalized as a sequence of M *DSS*, for instance by a simple greedy *DSS* segmentation algorithm or with the Minimum Perimeter Polygon [13]. We then calculate the characteristics of each *DSS* when changing the resolution of the grid (see [14] for more details about the covering of a DSL). In this case, a faster computation of the *DSS* parameters is possible.

Many works deal with the relations between irreducible rational fractions and digital lines (see [4] for characterization with Farey series, and [18] for a link with decomposition into continuous fractions). In [2], Debled and Réveillès first introduced the link between the Stern-Brocot tree and the recognition of digital line. Recognizing a piece of digital line is like going down the Stern-Brocot tree up to the directional vector of the line. To sum up, the classical online *DSS* recognition algorithm **DR95** [2] (also reported in [8]) updates the *DSS* slope when adding a point that is just exterior to the current line (weak exterior points). In [17], De Vieilleville and Lachaud have revisited a classical arithmetically-based *DSS* recognition algorithm with new parameters related to a combinatorial representation of *DSS*. New analytic relations have been established and the relation with the Stern-Brocot tree has been made explicit.

The main contribution of this paper is to present a fast algorithm which computes the exact (minimal) characteristics of a *DSS* that is some subset of a DSL of known characteristics (see [15] for more details about minimal characteristics). More precisely, the input DSL, say D , is given as the continued fraction of its slope. The *DSS* is specified by the positions of its two endpoints A and B . Furthermore, the two lower leaning points of D surrounding A and B are given as input. This new algorithm, called **ReversedSmartDSS**¹, determines the characteristics of the *DSS* by moving in a bottom-up way along the Stern-Brocot tree, a famous tree structure that represents positive fractions. We prove the correctness of this algorithm in Proposition 1. We further show in Proposition 2 that its worst-case computational complexity is $\Theta(k - k')$, where $[u_0; u_1, \dots, u_k]$ is the continued fraction of the slope of the input DSL and $[u_0; u_1, \dots, u_{k'}]$ is the continued fraction of the slope of the output *DSS*. This result assumes a computing model where standard arithmetic operations are in $O(1)$, which is a reasonable assumption when analyzing digital shapes.

This complexity is first to compare with the **SmartDSS** algorithm [14], whose worst-case computational complexity is $\Theta(\sum_{i=0}^{k'} u_i * \delta)$, where δ is the number of patterns of the output *DSS*. The average of this sum for all fractions $\frac{a}{b}$ with $a + b = n$ is experimentally lower than $\log^2 n$, and this sum is upper bounded by n and lower bounded by 1. This complexity is secondly to compare with the complexities of classical *DSS* recognition algorithms (e.g., **DR95** [3], see

¹ This name is in opposition to the **SmartDSS** algorithm [14], because it moves along the Stern-Brocot in a reversed direction.

also [9]), whose complexities are at best $\Theta(n)$. The `ReversedSmartDSS` algorithm was implemented and tested and various experimental results show that this algorithm performs better than the `SmartDSS` algorithm and the classical DSS recognition algorithms (see Table 1).

Table 1. Computation times of the (h, v) -covering of various digital shapes with our proposed approach. The digital shapes are: a circle of radius 2000; a flower with 5 petals, mean radius 5000 and variability of radius 7000; a polygon with 8 sides and radius 2000. The symbol # stands for “number of”.

Shape	Flower			Circle			Polygon		
# points	67494			16004			15356		
# segments	1991			574			44		
h, v	2	4	10	2	4	10	2	4	10
# points (h, v)	33744	16870	6750	8000	4000	1600	7676	3840	1532
Smart DSS									
# points tested	19352	11254	4367	5413	2977	1019	782	667	527
timings (ms)	3.1286	2.6446	2.2914	0.997	0.8902	0.7618	0.1258	0.1142	0.0946
Reversed Smart DSS									
timings (ms)	2.364	2.103	2.078	0.758	0.702	0.625	0.104	0.097	0.084

In Section 2 we describe this new algorithm. We show the correctness of this new algorithm in Section 3, as well as its computational complexity. The last section concludes and presents future works.

2 A Coarsening Algorithm for Computing the Characteristics of a Subsegment Included in a Known DSL

We recall first that a standard *digital straight line* D in the fourth quadrant is some subset of the digital plane $\{(x, y) \in \mathbb{Z}^2, \nu \leq \alpha x + \beta y < \nu + \alpha + \beta\}$, where $(\alpha, \beta, \nu) \in \mathbb{Z}^+ \times \mathbb{Z}^+ \times \mathbb{Z}$ form the *characteristics* of D . The fraction $\frac{\alpha}{\beta}$ is the *slope* of the DSL. Any 4-connected piece of a DSL is a *digital straight segment* (DSS). The *characteristics* of a DSS are the characteristics of the “simplest” DSL covering it (the one with the smallest possible α). Lastly, a *pattern* (resp. *reversed pattern*) is the Freeman chaincode joining two consecutive lower leaning points (resp. upper leaning points). Patterns are a combinatorial characterization of DSL.

In the next section, we describe our new algorithm for determining the characteristics of any subsegment S of a digital straight line D in the fourth quadrant, whose characteristics (α, β, ν) are known. We make use of the property that the slope of S is either $\frac{\alpha}{\beta}$ or any one of the ancestors of $\frac{\alpha}{\beta}$ in the Stern-Brocot tree ([15], see also Proposition 3 of [14]). The principle of our new algorithm is to follow a bottom-up way in the Stern-Brocot tree.

2.1 Overview of the Algorithm

Algorithm 1 is the general algorithm for computing the exact characteristics of S knowing the characteristics (α, β, μ) of its covering line D . This algorithm

```

Function ReversedSmartDSS (In  $D : DSL(\alpha, \beta)$ , In  $L_1, L_2 : \text{Points of } \mathbb{Z}^2$ , In  $A, B : \text{Points of } \mathbb{Z}^2$ ) :  $DSS(a, b, \mu)$ ;
Var  $Lp_1, Lp_2 : \text{Point of } \mathbb{Z}^2$ ;
Var  $dL : \text{integer}$  /* The horizontal distance between  $L_1$  and  $L_2$  */;
Var  $S : \text{slope}$ ;
begin
1  if ( $A_x == B_x$ ) then return  $(1, 0, A_x)$ ;
2  if ( $A_y == B_y$ ) then return  $(0, 1, A_y)$ ;
    $dL \leftarrow L_{2x} - L_{1x}$ ;
3  if ( $dL \geq 3\beta$ ) or ( $dL == 2\beta$  and ( $A == L_1$  or  $B == L_2$ )) or ( $A == L_1$ 
   and  $B == L_2$ ) then return  $(\alpha, \beta, \alpha L_{1x} + \beta L_{1y})$ ;
   /*  $S$  is included in two patterns of  $D$  */;
4  if ( $dL == 2\beta$ ) then return DSSWithinTwoPatterns ( $D, L_1, L_2, A, B$ );
   /*  $S$  is included in one pattern of  $D$  */;
5  ( $D(\alpha', \beta'), Lp_1, Lp_2$ )  $\leftarrow$  NewLowerBound( $D, L_1, L_2, A, B$ );
6  if ( $Lp_1 == L_1$ ) and ( $Lp_2 == L_2$ ) then
   | return FinalSlope( $D, L_1, L_2, Lp_1, Lp_2, A, B$ );
7  return ReversedSmartDSS( $DSL(\alpha', \beta'), Lp_1, Lp_2, A, B$ );
end

```

Algorithm 1. Main algorithm. Computes the characteristics (a, b, μ) of a DSS S that is some subset of a DSL D of slope $\frac{\alpha}{\beta}$ and lower leaning points L_1 and L_2 (the ones surrounding the segment AB). The DSS is defined by a starting point A and an ending point B ($A, B \in D$).

thus computes the simplest DSL covering S . The segment S is defined by its two endpoints A and B . Lastly, we give also as input the two lower leaning points of D which surround A and B . Note that these input data are all known if the DSL D was recognized by a classical recognition algorithm (e.g., DR95 [3]).

If A and B have the same abscissa (or same ordinate), then this algorithm stops and return $(1, 0, A_x)$ (or $(0, 1, A_y)$), which are the obvious results. Otherwise, this algorithm then checks the horizontal distance between L_1 and L_2 to measure the number of patterns covering the segment. Should this distance be: (1) three times greater than β , (2) equal to 2β , A and L_1 are superposed, or B and L_2 are superposed, or (3) A and L_1 are superposed, and B and L_2 are superposed, then the algorithm stops and returns (α, β) (line 3). Indeed, in these case, the DSS contains at least one pattern, so the characteristics follow.

Otherwise, if this horizontal distance is equal to 2β — S is included in two patterns of D — then the characteristics are computed by the function **DSSWithinTwoPatterns** (line 4), described by Algorithm 2.

2.2 S Is Included in Two Patterns of D

The function **DSSWithinTwoPatterns** (Algorithm 2) is really the core of this DSS recognition method. In its loop, three different patterns are tested progressively, so as to find the first that has exactly the sought slope. It is easy to see that, since the segment is included in two patterns, the middle lower leaning point L_m is the lowest point of the segment. Therefore, if the slope of the segment is defined by a pattern (i.e. defined by two lower leaning point), then one of the extremities of the pattern is this point L_m . We thus test in sequence the possible patterns to the left and to the right of L_m . However, the slope of a DSS may

also be defined by a reversed pattern (i.e. defined by two upper leaning points). We thus test also in sequence the possible reversed patterns.

The progressive computation of these three sequences of patterns (left pattern, right pattern, reversed pattern) is done in a parallel manner. More precisely, they are run consecutively one step at each time:

- line 4 with a call to `LowerSlope(..., true)` computes the next left patterns and stops when the lower leaning point L_1 overtakes or reaches A ,
- line 5 with a call to `LowerSlope(..., false)` computes the next right patterns and stops when the lower leaning point L_2 overtakes or reaches B ,
- line 6 with a call to `UpperSlope` computes the next reversed pattern and stops when the upper point RU overtakes towards the left or reaches B and the upper point LU overtakes towards the right or reaches A .

The conditions at lines 3, 7 and 8 corresponds to three possible cases where respectively no, one, or two pattern computation(s) is/are stopped. At each iteration, the pattern, reversed or not, with deepest slope is the candidate solution. Various tests of the positions of the current leaning points with respect to the segment AB allow to determine if this candidate is valid. If it is true, the function exits and returns the characteristics of the elected pattern. If it is false, the function loops and computes the new three patterns. This function loops at most k times, where k is the depth of the input slope.

Algorithm 3 computes in $O(1)$ the characteristics of the lower bound in the left (or right according to the boolean variable *Left*) of L_m . In this algorithm, we fix L_m , move L_1 right towards A (or L_2 left towards B in the other case) and calculate the value of the new slope between the new L_1 and L_m (or L_m and the new L_2).

Algorithm 4 determines in $O(1)$ the positions of the new upper leaning points LU and RU , according to the parity of slope (line 13) (see 17 for more details about the parity of the slope). We have moved LU and RU either towards the left in the odd case or towards the right in the even case. But before moving, we must calculate the number of subpatterns, that is covering the point A in the even case (line 2) or B in the odd case (line 4).

Example. Let us look at a run of Algorithm 1 for the DSL D of slope $13/18 = [0, 1, 2, 1, 1, 2]$ (Fig. 1), for the subsegment AB . Here the segment S is included in two patterns $13/18$. Since the condition on line 3 of Algorithm 1 is fulfilled, we call `DSSWithinTwoPatterns` (Algorithm 2) to compute the characteristics (a, b, μ) of S .

For the first step, left pattern has slope $3/4$ (call `LowerSlope`, see Fig. 2a), right pattern has slope $5/7$ (call `LowerSlope`, see Fig. 2c), and reversed pattern has slope $5/7$ (call `UpperSlope`, see Fig. 2b). As Algorithm 4 is stopped, we thus compute the deepest slope of $3/4$, $5/7$ and $5/7$, which is $5/7$. Since the deepest slope coincides with the slope returned by the stopped algorithm, Algorithm 2 stops and returns this slope (final result $(5, 7, 6)$).

Let us now give some explanations of (Fig. 2a). In the first step, we fix L_{11} at L_1 and L_{22} at L_m , and we compute the previous slope $PS = 5/7$ of $S = 13/18$. Since the parity of S is odd, then we calculate the number k of subpatterns


```

Function DSSWithinTwoPatterns( In  $D$  : DSL  $(\alpha, \beta)$ , In  $L_1, L_2$  : Lower
bounds of  $D$ , In  $A, B$  : Point of  $\mathbb{Z}^2$ ) : DSS  $(a, b, \mu)$ ;
Var  $U_1, U_2, L_m$  : Point of  $\mathbb{Z}^2$ ;  $L, R, U$  : boolean;
Var  $S_1, S_2, S_3, DS$  : slope;
Var  $D_L, D_R, D_U$  : DSL  $(\alpha, \beta)$ ,  $CF$  : Continued Fraction of  $D$ ;
begin
   $L \leftarrow \text{true}, R \leftarrow \text{true}, U \leftarrow \text{true}, i \leftarrow 0$ ;
1   $L_m \leftarrow \text{MiddleLowerBound}(L_1, D), CF \leftarrow \text{ContinuedFraction}(D)$ ;
2   $U_1 \leftarrow \text{FirstUpperBound}(D, L_1, L_2), U_2 \leftarrow \text{SecondUpperBound}(U_1, D)$ ;
  while  $i < |CF|$  do
3    if  $(L \text{ and } R \text{ and } U)$  then
4       $S_1 \leftarrow \text{LowerSlope}(D_L, L_1, L_m, L_2, A, \text{true})$  /* Left Lower Slope */;
5       $S_2 \leftarrow \text{LowerSlope}(D_R, L_1, L_m, L_2, B, \text{false})$  /* Right Lower Slope */;
6       $S_3 \leftarrow \text{UpperSlope}(D_U, U_1, U_2, A, B)$  /* Upper Slope */;
      if  $(L_1 \geq A \text{ or } L_2 \leq B \text{ or } (U_1 \geq A \text{ and } U_2 \leq B))$  then
         $DS \leftarrow \text{DeepestSlope}(S_1, S_2, S_3)$  /* DS The deepest slope */;
        if  $L_1 \geq A$  then  $L \leftarrow \text{false}$ ;
        if  $L_2 \leq B$  then  $R \leftarrow \text{false}$ ;
        if  $(U_1 \geq A \text{ and } U_2 \leq B)$  then  $U \leftarrow \text{false}$ ;
        if  $(L_1 \geq A \text{ and } DS == S_1) \text{ or } (L_2 \leq B \text{ and } DS == S_2) \text{ or}$ 
           $(U_1 \geq A \text{ and } U_2 \leq B \text{ and } DS == S_3)$  then break;
7    else if  $((L \text{ and } R) \text{ or } (L \text{ and } U) \text{ or } (R \text{ and } U))$  then
       $S_1 \leftarrow (R \text{ and } U) ? \text{LowerSlope}(D_R, L_1, L_m, L_2, B, \text{false})$ 
        :  $\text{LowerSlope}(D_L, L_1, L_m, L_2, A, \text{true})$ ;
       $S_2 \leftarrow (L \text{ and } R) ? \text{LowerSlope}(D_R, L_1, L_m, L_2, B, \text{false})$ 
        :  $\text{UpperSlope}(D_U, U_1, U_2, A, B)$ ;
      if  $((((L \text{ and } U) \text{ or } (L \text{ and } R)) \text{ and } (L_1 \geq A)) \text{ or } (((R \text{ and } U) \text{ or}$ 
         $(L \text{ and } R)) \text{ and } (L_2 \leq B)) \text{ or } (((R \text{ and } U) \text{ or } (L \text{ and } U)) \text{ and}$ 
         $(U_1 \geq A \text{ and } U_2 \leq B)))$  then  $DS \leftarrow \text{DeepestSlope}(S_1, S_2)$ ;
      if  $((((R \text{ and } U) \text{ and } ((L_2 \leq B \text{ and } DS == S_1) \text{ or } (U_1 \geq A \text{ and}$ 
         $U_2 \leq B \text{ and } DS == S_2))) \text{ or } ((L \text{ and } U) \text{ and } ((L_1 \geq A \text{ and}$ 
         $DS == S_1) \text{ or } (U_1 \geq A \text{ and } U_2 \leq B \text{ and } DS == S_2))) \text{ or } ((L$ 
         $\text{ and } R) \text{ and } ((L_1 \geq A \text{ and } DS == S_1) \text{ or } (L_2 \leq B \text{ and}$ 
         $DS == S_2))))$  then break;
      if  $((((L \text{ and } U) \text{ or } (L \text{ and } R)) \text{ and } (L_1 \geq A))$  then  $L \leftarrow \text{false}$ ;
      if  $((((R \text{ and } U) \text{ or } (L \text{ and } R)) \text{ and } (L_2 \leq B))$  then  $R \leftarrow \text{false}$ ;
      if  $((((R \text{ and } U) \text{ or } (L \text{ and } U)) \text{ and } (U_1 \geq A \text{ and } U_2 \leq B))$  then
         $U \leftarrow \text{false}$ ;
8    else
      if  $L$  then  $DS \leftarrow \text{LowerSlope}(D_L, L_1, L_m, L_2, A, \text{true})$ ;
      else if  $R$  then  $DS \leftarrow \text{LowerSlope}(D_R, L_1, L_m, L_2, B, \text{false})$ ;
      else  $DS \leftarrow \text{UpperSlope}(D_U, U_1, U_2, A, B)$ ;
      if  $((L \text{ and } L_1 \geq A) \text{ or } (R \text{ and } L_2 \leq B) \text{ or } (U \text{ and } U_1 \geq A \text{ and}$ 
         $U_2 \leq B))$  then break;
     $a \leftarrow DS_x, b \leftarrow DS_y, \mu \leftarrow aL_{m_x} + bL_{m_y}$ ;
  return  $(a, b, \mu)$ ;
end

```

Algorithm 2. Computes the characteristics (a, b, μ) of a DSS S that is some subset of a DSL $D(\alpha, \beta)$ repeated twice

```

Function LowerSlope( In  $D : \text{DSL}(\alpha, \beta)$ , InOut  $L_1, L_m, L_2 : \text{Lower bounds of } D$ , In  $X(A$ 
or  $B) : \text{Point of } \mathbb{Z}^2$ , In  $\text{Left} : \text{Boolean}$ ):  $\text{DSS}(a, b)$ ;
Var  $P, L_{11}, L_{22} : \text{Points of } \mathbb{Z}^2$  /*  $L_{11}$  and  $L_{22}$  two lower leaning points */;
Var  $PS : \text{slope}$ ;
Var  $k : \text{integer}$ ;
Var  $\text{parity} : \text{boolean}$ ;
begin
   $(P, L_{11}, L_{22}) \leftarrow \text{Left} ? (L_1, L_1, L_m) : (L_2, L_m, L_2)$ ;
   $\text{parity} \leftarrow \text{Parity}(D)$ ;
   $PS \leftarrow \text{PreviousSlope}(D)$ ;
  if ( $\text{parity}$  is odd) then
     $k \leftarrow \text{NumberOfCoveringSubPatterns}(L_{11}, X, PS, \text{true}, \text{false})$ ;
     $P \leftarrow L_{11} - k(-PS_y, PS_x)$ ;
  else
     $k \leftarrow \text{NumberOfCoveringSubPatterns}(L_{22}, X, PS, \text{true}, \text{false})$ ;
     $P \leftarrow L_{22} - k(PS_y, -PS_x)$ ;
  if  $\text{Left}$  then  $L_{11} = P$ ;
  else  $L_{22} = P$ ;
   $(a, b) \leftarrow (|P_y - L_{m_y}|, |L_{m_x} - P_x|)$ ;
   $(L_1, L_2) \leftarrow \text{Left} ? (L_{11}, L_2) : (L_1, L_{22})$ ;
  return  $(a, b)$ ;
end

```

Algorithm 3. Computes in $O(1)$ the Lower (Left or Right) characteristics (a, b) of a DSS that is some subset of a DSL D , given a starting point A and an ending point L_m (Left part) or given a starting point L_m and an ending point B (Right part) ($A, B \in D$) (Left pattern is $L_1 L_m$ and Right pattern is $L_m L_2$)

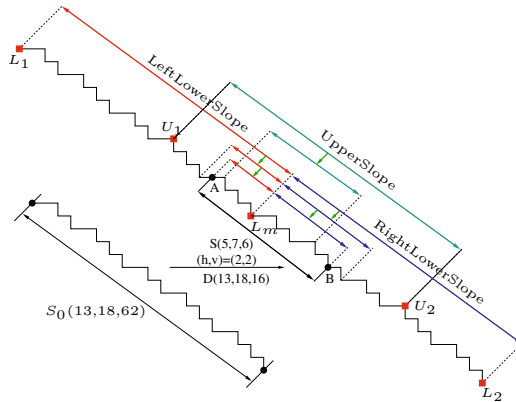


Fig. 1. A DSL $D(13, 18, 16)$ with two patterns between L_1 and L_2 and an odd depth slope. $S(AB)$ is the covering of S_0 by the tiling $(h, v) = (2, 2)$, and also a subset of D (AB is included in two patterns of D). Lower and upper leaning points are drawn as red boxes. The (red, blue or cyan) arrows represent bottom-up move along the Stern-Brocot Tree.

$5/7$ that is covering A from L_{11} to the right, such that this covering reaches or overtakes A . It is impossible to take $k = 3$, because in this case this displacement from L_{11} overtakes L_{22} . So we take $k = 2$, and L_{11} is moved of two subpatterns $5/7$ toward the right. We have now a new DSS $S(3, 4)$ between the new L_{11} and L_{22} . In the second step, we calculate the previous slope $PS = 1/1$ of $S = 3/4$. Since the parity of S is even, then we calculate the number k of subpatterns

```

Function UpperSlope( In  $D : \text{DSL}(\alpha, \beta)$ , In  $U_1, U_2$ : Upper bound of  $D$ , In  $A, B$  : Points of  $\mathbb{Z}^2$ ) :  $\text{DSS}(a, b)$ ;
Var  $LU, RU$  : Point of  $\mathbb{Z}^2$  /* Left and Right upper leaning points */;
Var  $S, PS$  : slope /*  $PS$  the previous slope of  $S$  */;
Var  $k$  : integer /* Number of subpatterns covering  $A$  or  $B$  */;
Var  $parity$  : boolean /* parity of slope continued fraction */;
begin
     $LU \leftarrow U_1, RU \leftarrow U_2$ ;
    if ( $LU > A$  and  $RU < B$ ) then
        return  $(\alpha, \beta)$ ;
     $parity \leftarrow \text{Parity}(D), PS \leftarrow \text{PreviousSlope}(D)$ ;
1   if ( $parity$  is odd) then
2     if ( $RU > B$ ) then
         $k \leftarrow \text{NumberOfCoveringSubPatterns}(RU, B, PS, true, false)$ ;
         $RU \leftarrow RU - k(PS_y, -PS_x)$ ;
        if ( $LU < A$ ) then
             $LU \leftarrow RU - (PS_y, -PS_x)$ ;
3     else
4       if ( $LU < A$ ) then
             $k \leftarrow \text{NumberOfCoveringSubPatterns}(LU, A, PS, true, false)$ ;
             $LU \leftarrow LU - k(-PS_y, PS_x)$ ;
            if ( $RU > B$ ) then
                 $RU \leftarrow LU - (-PS_y, PS_x)$ ;
         $(a, b) \leftarrow (LU_y - RU_y, RU_x - LU_x)$ ;
    return  $(a, b)$ ;
end

```

Algorithm 4. Computes in $O(1)$ the Upper characteristics (a, b) of a DSS that is some subset of a DSL D (U_1 and U_2 are two upper leaning points of D), given a starting point A and an ending point B ($A, B \in D$)

1/1 that is covering A from L_{22} to the left, such that this covering reaches or overtakes A . It is impossible to take $k = 4$, because in this case this displacement from L_{22} overtakes L_{11} . So we take $k = 3$, and L_{11} is moved of three subpatterns 1/1 toward the left and we obtain a new DSS from the new L_{11} to L_{22} . As L_{11} reach A , this algorithms stops and returns the left slope 1/1.

2.3 S Is Included in One Pattern of D

In the remaining case, S is included in one pattern only. The function **New LowerBounds** (Algorithm 5) attempts to move L_1 toward A with k_1 subpatterns and L_2 toward B with k_2 subpatterns, according to the parity of the depth of the development of the slope in continued fractions (it means that $Parity(D)$ is true if D has a slope with even depth, false otherwise) and to the previous convergent of the slope (line 5) (if the depth of the slope is a $k - th$ convergent, then the previous slope is a $(k - 1) - th$ convergent). This displacement is constrained by the fact that the leaning points must still surround A and B (see Fig. 3(a),(b)). In the case where L_1 and L_2 did not move, we can conclude on the characteristics with a call to **FinalSlope** (Algorithm 6). In this algorithm, we focus on the three cases. Firstly, if the second lower leaning point Lp_2 is equal to B and Lp_1 moves toward A with either one previous slope or previous previous slope, according to the parity of the depth of the slope. Secondly, if the first lower leaning point Lp_1 is equal to A and Lp_2 moves towards B . Finally, if the lower leaning points

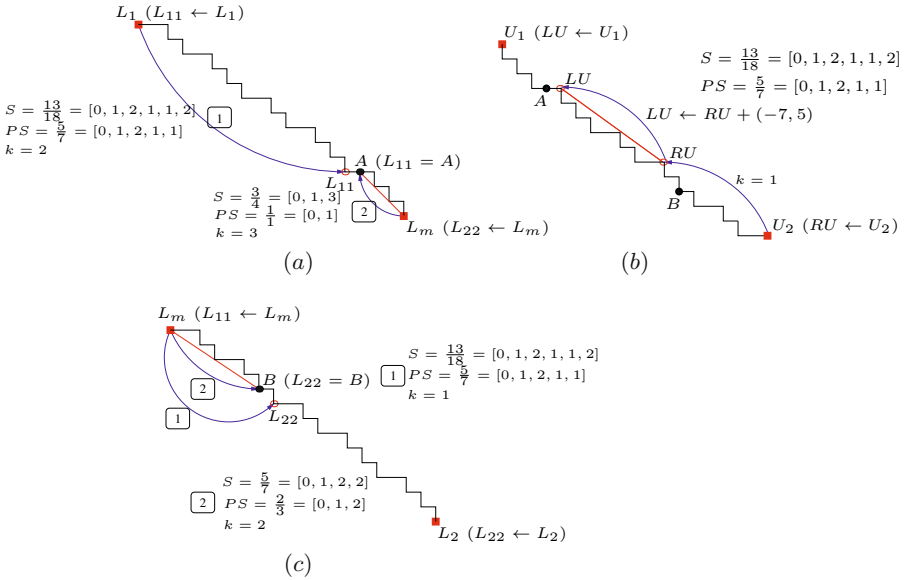


Fig. 2. The DSS U_1U_2 (resp. L_1L_m and L_mL_2) of characteristics $(13, 18, 16)$, which is a subset of L_1L_2 of Fig. 1. The blue arrows represent the move between the upper (lower) leaning points, and k is the number of subpatterns covering B (A and B).

```

Function NewLowerBounds( In  $D : \text{DSL}(\alpha, \beta)$ , In  $L_1, L_2, A, B : \text{Points of } \mathbb{Z}^2$ ) : ( $\text{DSL}$ ,  $\text{Point of } \mathbb{Z}^2$ ,  $\text{Point of } \mathbb{Z}^2$ );
Var  $L, V_1, V_2 : \text{Point of } \mathbb{Z}^2$ ;
Var  $k, k_1, k_2 : \text{integer}$ ;
Var  $\text{parity}, \text{covering}_A, \text{covering}_B : \text{boolean}$ ;
Var  $PS : \text{slope}$ ;
begin
   $\text{parity} \leftarrow \text{Parity}(D)$ ,  $PS \leftarrow \text{PreviousSlope}(D)$ ;
   $L \leftarrow \text{parity} ? L_1 : L_2$ ,  $\text{covering}_A \leftarrow \text{parity} ? \text{true} : \text{false}$ ;
   $\text{covering}_B \leftarrow \text{parity} ? \text{false} : \text{true}$ ;
   $k_1 \leftarrow \text{NumberOfCoveringSubPatterns}(L, A, PS, \text{covering}_A, \text{true})$ ;
   $k_2 \leftarrow \text{NumberOfCoveringSubPatterns}(L, B, PS, \text{covering}_B, \text{true})$ ;
  if ( $\text{parity}$  is odd) then
     $Lp_1 \leftarrow L_1 - k_1(-PS_y, PS_x)$ ,  $V_2 \leftarrow L_1 - k_2(-PS_y, PS_x)$ ;
     $Lp_2 \leftarrow (V_2 \leq L_2) ? V_2 : L_2$ ;
  else
     $Lp_2 \leftarrow L_2 - k_2(PS_y, -PS_x)$ ,  $V_1 \leftarrow L_2 - k_1(PS_y, -PS_x)$ ;
     $Lp_1 \leftarrow (V_1 \geq L_1) ? V_1 : L_1$ ;
   $k \leftarrow (\text{parity}$  is odd and  $Lp_2! = L_2$ ) or ( $\text{parity}$  is even and  $Lp_1! = L_1$ ) ?
   $(Lp_{2x} - Lp_{1x})/PS_y : 1$ ;
   $(\alpha, \beta) \leftarrow ((Lp_{1y} - Lp_{2y})/k, (Lp_{2x} - Lp_{1x})/k)$ ;
  return ( $\text{DSL}(\alpha, \beta)$ ,  $Lp_1, Lp_2$ );
end

```

Algorithm 5. Updates in $O(1)$ the slope of the DSL $D(\alpha, \beta)$ according to the change of the two lower leaning points from (L_1, L_2) to (Lp_1, Lp_2)

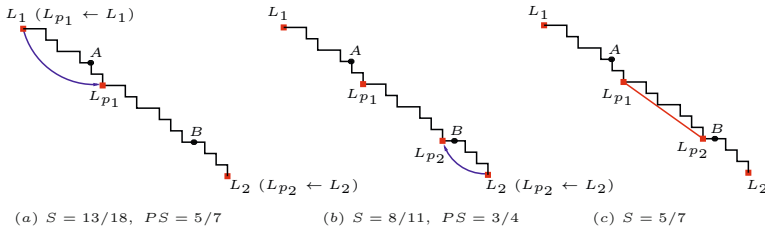


Fig. 3. L_1L_2 is a DSS of characteristics $(13, 18, 16)$. The blue arrows represent the move of the lower leaning points. (a) $S = L_1L_2$, L_1 is moved of one subpatterns $5/7$ towards the right. (b) $S = L_{p_1}L_2$, L_2 is moved of one subpatterns $3/4$ towards the left. (c) L_{p_1} and L_{p_2} represent the lower leaning points of AB .

```

Function FinalSlope( In  $D : \text{DSL}(\alpha, \beta)$ , In  $L_1, L_2, L_{p_1}, L_{p_2}, A, B : \text{Points of } \mathbb{Z}^2$ ) : DSS
( $a, b, \mu$ );
Var  $PS, PPS$  : slope;
Var  $parity$  : boolean;
begin
   $PS \leftarrow \text{PreviousSlope}(D)$ ,  $PPS \leftarrow \text{PreviousSlope}(PS)$ ,  $parity \leftarrow \text{Parity}(D)$ ;
  if ( $L_{p_2} == B$ ) then
     $L_{p_1} \leftarrow L_{p_1} - ((parity) ? (-PS_y, PS_x) : (-PPS_y, PPS_x))$ ;
    ( $a, b$ )  $\leftarrow (L_{p_1}_y - L_{p_2}_y, L_{p_2}_x - L_{p_1}_x)$ ,  $\mu \leftarrow aL_{2x} + bL_{2y}$ ;
  else if ( $L_{p_1} == A$ ) then
     $L_{p_2} \leftarrow L_{p_2} - ((parity) ? (PPS_y, -PPS_x) : (PS_y, -PS_x))$ ;
    ( $a, b$ )  $\leftarrow (L_{p_1}_y - L_{p_2}_y, L_{p_2}_x - L_{p_1}_x)$ ,  $\mu \leftarrow aL_{1x} + bL_{1y}$ ;
  else
    ( $a, b$ )  $\leftarrow PS$ ;
     $\mu \leftarrow a(parity ? L_{1x} : L_{2x}) + b(parity ? L_{1y} : L_{2y})$ ;
  return ( $a, b, \mu$ );
end

```

Algorithm 6. Computes in $O(1)$ the characteristics (a, b, μ) of a DSS in the case where $L_1 == L_{p_1}$ and $L_2 == L_{p_2}$

did not move. We therefore conclude that AB has the slope of $L_{p_1}L_{p_2}$ (see Fig. 3,(c)). If at least one of the leaning points has moved, then we recursively call `ReversedSmartDSS` but with an input DSL with a slope depth at least one smaller than the slope depth of D .

3 Correctness and Computational Complexity

We sketch here the proof of the correctness of algorithm `ReversedSmartDSS` (Proposition 1). Proposition 2 establishes the time complexity of this algorithm, as a function of the depth of the slopes of the input DSL and output DSS.

Proposition 1. For any DSL D such that $A, B \in D$, Algorithm 1 computes the characteristics of the segment $S = [AB]$ included in D .

Proof. We prove by induction on n (the depth of the slope of D) that Algorithm 1 computes the characteristics of the segment S in the fourth quadrant. The initial steps $n = -1$ or $n = 0$ are obvious since D is just a vertical or horizontal

segment and has slope $1/0$ or $0/1$. In this case, Algorithm 1 returns the correct characteristics $(1, 0, A_x)$ or $(0, 1, A_y)$. The induction hypothesis is that this algorithm has a correct output for all points $A, B \in D$ of slope $[u_0, u_1, \dots, u_n]$. We shall prove that the output is also correct for any points A, B in some DSL D with a depth of its slope equal to $n + 1$.

If the horizontal distance between the lower leaning points L_1 and L_2 verifies one of the three conditions of line 3, then Algorithm 1 stops and returns the characteristics of the segment AB which are trivially in this case the characteristics of D itself (i.e. depth is $n + 1$).

Otherwise, if this distance is equal to 2β , then we call Algorithm 2. There, three algorithms are run in a parallel manner: (i) `LowerSlope`(..., true) at line 4 for the pattern to the left of L_m , (ii) `LowerSlope`(..., false) at line 5 for the pattern to the right of L_m , and (iii) `UpperSlope` at line 6 for the reversed pattern containing L_m . More precisely, they are run consecutively one iteration each time. This algorithm does not depend on the induction hypothesis. It just calculates the largest pattern contained in $[AB]$ either to the left or right of the lowest point L_m , and the largest reversed pattern containing L_m . Since patterns characterize a DSS slope, the deepest slope is exactly the slope of AB .

The last case occurs when S is included in only one pattern of D . We then look for a simpler DSL than D that contains AB by moving the lower leaning points toward AB (function `NewLowerBounds`). The simpler DSL is one of the left or right ancestor of the slope of D in the Stern-Brocot tree, according to its parity. If such a simpler DSS includes AB , then the lower leaning points are moved. Here, we recursively call `ReversedSmartDSS` but with this new input data, where the input DSL has a slope depth less or equal to n . The induction hypothesis applies in this case. If no simpler DSS contains AB , the function `FinalSlope` determines directly (without loop) the correct characteristics, with simple checks. We can therefore conclude in all cases. □

We assume that we have stored all the convergents of the slope of D before running Algorithm 1. We further assume a computing model where standard arithmetic operations takes $O(1)$. Note that the largest integer used in the presented algorithms is lower than $\alpha^2 + \beta^2$, if D has slope $\frac{\alpha}{\beta}$, for a frame centered on the DSS.

Proposition 2. *Algorithm 1 takes $O(n - n')$ time complexity, where n is the depth of the input DSL D with slope $\frac{\alpha}{\beta} = [u_0, u_1, \dots, u_n]$ and n' is the depth of the output DSS S with slope $\frac{a}{b} = [u_0, u_1, \dots, u_{n'}]$.*

Proof. Computation of Algorithm 1 on line 1, 2 and 3 is clearly $O(1)$. Otherwise, if S is included in two patterns of D (line 4), we then apply Algorithm 2 (detailed in the previous proposition). In the core of this algorithm, we call three algorithms where the time complexity of each algorithm is $O(1)$. Furthermore, we repeat either the same condition (line 3) until one of the three algorithms is stopped, line 7 if one is stopped, or line 8 if two are stopped. Since the stopping occurs when the output slope is reached, the number of steps of

Algorithm 2 is the difference between the depth slope of the input DSL and the one of the output DSS, that is $O(n - n')$.

Otherwise, it means that S is included in one pattern of D . In this case, we compute in $O(1)$ the positions of the new two lower leaning points L_1 and L_2 of D by calling `NewLowerBounds` (line 5). When the lower leaning points do not move, then the function `FinalSlope` determines the correct characteristics in $O(1)$. Otherwise, we recursively call `ReversedSmartDSS` but with an input DSL with a slope depth less or equal to $n - 1$. The preceding arguments recursively hold. With this observation, the worst-case computational complexity of Algorithm 1 is clearly $O(n - n')$. \square

Lamé's theorem implies that Algorithm 1 takes at most $O(\log(\max(\alpha, \beta)))$ time.

Timing measures. Execution times were measured for some contours (Table 1). These times were obtained on a 2.10 GHz Intel Core 2 Duo. The listed numbers include the computation time for subsampling contours and the associated number of tested points in the `SmartDSS` algorithm [14] and the computation of the characteristics of each segment in the `ReversedSmartDSS` algorithm.

4 Conclusion

We have presented a novel fast DSS recognition algorithm with guaranteed logarithmic complexity, in the special case where a DSL container is known. The algorithm principle is to move in a bottom-up way along the Stern Brocot Tree, starting from the initial known DSL slope. Finally, we have used this algorithm to efficiently compute the exact multiscale covering of a digital contour (Table 1). Our algorithms are sensitive to the depth of the input DSL and output DSS, and are clearly sublinear.

References

1. Anderson, T.A., Kim, C.E.: Representation of digital line segments and their preimages. *Computer Vision, Graphics, and Image Processing* 30(3), 279–288 (1985)
2. Debled-Rennesson, I.: Etude et reconnaissance des droites et plans discrets. Ph.D. thesis, Université Louis Pasteur, Strasbourg (1995)
3. Debled-Rennesson, I., Reveillès, J.P.: A linear algorithm for segmentation of discrete curves. *International Journal of Pattern Recognition and Artificial Intelligence* 9, 635–662 (1995)
4. Dorst, L., Smeulders, A.W.M.: Discrete representation of straight lines. *IEEE transactions Pattern Analysis Machine Intelligence* 6, 450–463 (1984)
5. Dorst, L., Smeulders, A.W.M.: Discrete straight line segments: Parameters, primitives and properties. In: *Vision Geometry, Series Contemporary Mathematics*, pp. 45–62. American Mathematical Society, Providence (1991)
6. Freeman, H.: On the encoding of arbitrary geometric configurations. *Transactions on Electronic Computer* 10(2), 260–268 (1961)
7. Freeman, H.: Computer processing of line-drawing images. *ACM Comput. Surv.* 6(1), 57–97 (1974)

8. Klette, R., Rosenfeld, A.: *Digital Geometry - Geometric Methods for Digital Picture Analysis*. Morgan Kaufmann, San Francisco (2004)
9. Klette, R., Rosenfeld, A.: Digital straightness—a review. *Discrete Applied Mathematics* 139(1-3), 197–230 (2004), <http://www.sciencedirect.com/science/article/B6TYW-49YD4PJ-4/2/8a755750eee9d6517adbff2f20ee7dc2>, the 2001 International Workshop on Combinatorial Image Analysis
10. Kovalevsky, V.: Applications of digital straight segments to economical image encoding. In: Ahronovitz, E. (ed.) *DGCI 1997*. LNCS, vol. 1347, pp. 51–62. Springer, Heidelberg (1997)
11. Kovalevsky, V.A.: New definition and fast recognition of digital straight segments and arcs. In: *International Conference on Pattern Analysis and Machine Intelligence*, pp. 31–34 (1990)
12. Kovalevsky, V.A., Fuchs, S.: Theoretical and experimental analysis of the accuracy of perimeter estimates. In: Forster, W., Ruwiedel, S. (eds.) *Robust Computer Vision*, pp. 218–242 (1992)
13. Montanari, U.: A note on minimal length polygonal approximation to a digitized contour. *Communications of the ACM* 13(1), 41–47 (1970)
14. Said, M., Lachaud, J.-O., Feschet, F.: Multiscale Discrete Geometry. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 118–131. Springer, Heidelberg (2009), <http://hal.archives-ouvertes.fr/hal-00413681/en/>
15. Sivignon, I., Dupont, F., Chassery, J.M.: Digital intersections: minimal carrier, connectivity, and periodicity properties. *Graphical Models* 66(4), 226–244 (2004)
16. Troesch, A.: Interprétation géométrique de l’algorithme d’euclide et reconnaissance de segments. *Theor. Comput. Sci.* 115(2), 291–319 (1993)
17. de Vieilleville, F., Lachaud, J.O.: Revisiting digital straight segment recognition. In: Kuba, A., Palágyi, K., Nyúl, L.G. (eds.) *DGCI 2006*. LNCS, vol. 4245, pp. 355–366. Springer, Heidelberg (2006), <http://www.lama.univ-savoie.fr/~lachaud/Publications/LACHAUD-JO/publications.html#deVieilleville06a>
18. Yaacoub, J.: *Enveloppes convexes de réseaux et applications au traitement d’images*. Ph.D. thesis, Université Louis Pasteur, Strasbourg (1997), <http://lsiit.u-strasbg.fr/Publications/1997/Yaa97>

A Near-Linear Time Guaranteed Algorithm for Digital Curve Simplification under the Fréchet Distance

Isabelle Sivignon

`gipsa-lab`, CNRS, UMR 5216, F-38420, France
`isabelle.sivignon@gipsa-lab.grenoble-inp.fr`

Abstract. Given a digital curve and a maximum error, we propose an algorithm that computes a simplification of the curve such that the Fréchet distance between the original and the simplified curve is less than the error. The algorithm uses an approximation of the Fréchet distance, but a guarantee over the quality of the simplification is proved. Moreover, even if the theoretical complexity of the algorithm is in $\mathcal{O}(n \log(n))$, experiments show a linear behaviour in practice.

1 Introduction

Given a polygonal curve, the curve simplification problem consists in computing another polygonal curve that (i) approximates the original curve, (ii) satisfies a given error criterion, (iii) with as few vertices as possible. This problem arises in a wide range of applications, such as geographic information systems (GIS), computer graphics or computer vision, where the management of the level of details is of crucial importance to save memory space or to speed-up analysis algorithms.

This problem has been studied for many years for various metrics: classical L_1, L_2, L_∞ metrics (see [4] for a survey on simplification algorithms using these metrics), Hausdorff distance or Fréchet distance. While the L norms and Hausdorff distance are relevant measures for many applications, they do not always reflect the similarity or dissimilarity of two polygonal curves (see for instance the example given in [10] for the Hausdorff distance). The main reason of this discrepancy is that these metrics consider the curves as sets of points, and do not reflect the course of the curves. However, the course of the curve may be important in some applications, like handwriting recognition for instance [11]. The Fréchet distance is often used to overcome this problem as it nicely measures the similarity between two curves. The Fréchet distance can be intuitively defined considering a man walking his dog. Each protagonist walks along a path, and controls its speed independently, but cannot go backwards. The Fréchet distance between the two pathes is the minimal length of the leash required.

1.1 Fréchet Distance

Given two curves f and g specified by functions $f : [0, 1] \rightarrow \mathbb{R}^2$ and $g : [0, 1] \rightarrow \mathbb{R}^2$, and two non-decreasing continuous functions $\alpha : [0, 1] \rightarrow [0, 1]$ and $\beta :$

$[0, 1] \rightarrow [0, 1]$ with $\alpha(0) = 0, \alpha(1) = 1, \beta(0) = 0, \beta(1) = 1$, the *Fréchet distance* $\delta_F(f, g)$ between two curves f and g is defined as

$$\delta_F(f, g) = \inf_{\alpha, \beta} \max_{0 \leq t \leq 1} d(f(\alpha(t)), g(\beta(t)))$$

where d denotes the Euclidean distance. The polygonal curve simplification problem was first studied for the Fréchet distance by Godau [7]. Alt and Godau proposed in [3] an $\mathcal{O}(mn)$ -time algorithm to determine whether $\delta_F(P, Q) \leq \varepsilon$ for two polygonal curves P and Q of size n and m , and a given error $\varepsilon > 0$. The complexity turns out to be $\mathcal{O}((m^2n + n^2m) \log(mn))$ for the actual computation of the Fréchet distance between two curves. A recent work [6] proposes a near-linear time algorithm to compute an approximation of this distance.

1.2 Curve Simplification Problem

In the rest of the paper, we follow the notations used in [2] or [1]. Given a polygonal curve $P = \langle p_1, \dots, p_n \rangle$, a curve $P' = \langle p_{i_1}, \dots, p_{i_k} \rangle$ with $1 = i_1 < \dots < i_k = n$ is said to *simplify* the curve P . $P(i, j)$ denotes the subpath from p_i to p_j .

Given a pair of indices $1 \leq i \leq j \leq n$, $\delta_F(p_i p_j, P)$ denotes the error of the segment $p_i p_j$ with respect to $P(i, j)$. Then,

$$\delta_F(P', P) = \max_{1 \leq j \leq k} \delta_F(p_{i_j} p_{i_{j+1}}, P)$$

For the sake of clarity, the simplified notation $error(i, j) = \delta_F(p_i p_j, P)$ will sometimes be used. We also say that $p_i p_j$ is a *shortcut*.

P' is an ε -*simplification* of P if $\delta_F(P', P) \leq \varepsilon$. The optimization problem is the following: given a polygonal curve P , find a ε -simplification P' of P with the minimum number of vertices. The approach of Imai and Iri [8] leads to an optimal solution under the Fréchet distance in $\mathcal{O}(n^3)$.

In [2], the authors propose an $\mathcal{O}(n \log(n))$ algorithm. The base of their algorithm is greedy and very simple: points are added one by one while the error of the shortcut is lower than ε , otherwise the process starts over from the last point processed. The strength of their approach lies in the following property:

Lemma 1. [2, Lemma 3.3] *Let $P = \{p_1, p_2, \dots, p_n\}$ be a polygonal curve in \mathbb{R}^2 . For $1 \leq i \leq l \leq r \leq j \leq n$, $error(l, r) \leq 2error(i, j)$*

This means that for any shortcut $p_l p_r$ such that $error(l, r) > \varepsilon$, there does not exist a shortcut $p_i p_j$ such that $error(i, j) \leq \frac{\varepsilon}{2}$. They derive the following theorem:

Theorem 1. [2, Theorem 3.4] *Given a polygonal curve P in \mathbb{R}^d and a parameter $\varepsilon > 0$, we can compute in $\mathcal{O}(n \log(n))$ an ε -simplification P' of P under the Fréchet metric error with at most the number of vertices of an optimal $\frac{\varepsilon}{2}$ -simplification.*

The $\mathcal{O}(n \log(n))$ complexity is achieved with a dichotomic process in the greedy algorithm. The question we arouse in this paper is the following: does there exist

a linear-time guaranteed algorithm to compute an ε -simplification? We propose a guaranteed algorithm for digital curves whose complexity is $\Omega(n \log(n))$ in theory but behaves in $\mathcal{O}(n)$ in practice. The main features of this algorithm are the following: (i) approximation of the Fréchet distance as in [1], (ii) restriction to digital curves. In section 2 we present the approximated distance, and give the algorithm outline as a novel way of using the results of [1]. Section 3 is the core of the paper and details the approximated distance efficient and online update, with a restriction to digital curves to achieve a near-linear complexity. Section 4 begins with a theoretical study of the complexity and the guarantee of the proposed algorithm, and ends with some experimental results.

2 Guaranteed Algorithm Using an Approximated Distance

2.1 Approximating the Fréchet Distance

In [1], Ali Abam et al. study the following problem: given a possibly infinite sequence of points defining a polygonal path, maintain in a streaming setting, a simplification of this set of points with $2k$ points and a bounded error. The error is measured using the Hausdorff distance or the Fréchet distance. The Fréchet distance being computationally too costly for this framework, they show that $error(i, j)$ can be upper and lower bounded by functions of two values, namely $\omega(i, j)$ and $b(i, j)$. $\omega(i, j)$ is the width of the points of $P(i, j)$ in the direction $p_i p_j$. $b(i, j)$ is the length of the longest backpath in the direction $\overrightarrow{p_i p_j}$. Its precise definition requires the following other definitions:

Definition 1. Let l be a straight line of directional vector \vec{d} . α is the angle between l and the abscissa axis.

$proj_\alpha(p)$ denotes the orthogonal projection of p onto the line of angle α .

If $\overrightarrow{p_l p_m} \cdot \vec{d} < 0$, then $proj_\alpha(p_l)$ is “after” $proj_\alpha(p_m)$ in direction α , and we denote $proj_\alpha(p_l) \gg proj_\alpha(p_m)$ (see Figure 1(a)).

Definition 2. $\overrightarrow{p_l p_m}$ is a positive shift if and only if $\overrightarrow{p_l p_m} \cdot \vec{d} > 0$, negative otherwise.

Definition 3. A backpath in direction α is a negative shift $\overrightarrow{p_l p_m}$ such that $l < m$ (see Figure 1(b)). p_l is the origin of the backpath. The length of the backpath is equal to $d(proj_\alpha(p_l), proj_\alpha(p_m))$.

Lemma 2 relates $\omega(i, j)$ and $b(i, j)$ to $error(i, j)$:

Lemma 2. [1, Lemma 4.2] The Fréchet error of a shortcut $p_i p_j$ satisfies $\max(\frac{\omega(i,j)}{2}, \frac{b(i,j)}{2}) \leq error(i, j) \leq 2\sqrt{2} \max(\frac{\omega(i,j)}{2}, \frac{b(i,j)}{2})$.

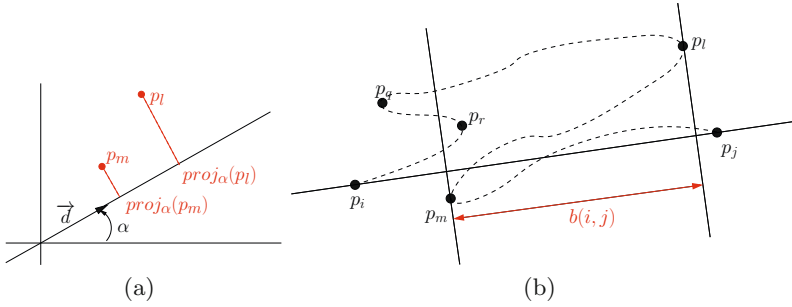


Fig. 1. (a) Illustration of Definitions 1 and 2. (b) Illustration of backpaths in direction $\overrightarrow{p_i p_j}$: $\overrightarrow{p_r p_q}$ is a backpath since it is a negative shift and p_r is before p_q on the curve $P(i, j)$. However, it is not the longest backpath: $\overrightarrow{p_i p_m}$ is also a backpath, of maximal length $b(i, j)$.

Algorithm 1. Greedy Fréchet simplification algorithm

```

i = 1, j = 2
while i < n do
    while j < n and max(w(i, j), b(i, j)) ≤ ε/√2 do
        j = j + 1
    create a new shortcut p_i p_{j-1}
    i = j - 1, j = i + 1

```

2.2 Algorithm Outline

Algorithm 1 presents the general outline of the ϵ -simplification.

Lemma 3. Algorithm 1 computes an ϵ -simplification P' of a polygonal curve P such that $|P'|$ is lower than the number of vertices of an optimal $\frac{\epsilon}{4\sqrt{2}}$ -simplification of P .

Proof. When a shortcut $p_i p_j$ is created in P' , the following two properties are true: (i) $\max(w(i, j), b(i, j)) \leq \frac{\epsilon}{\sqrt{2}}$ and (ii) $\max(w(i, j + 1), b(i, j + 1)) > \frac{\epsilon}{\sqrt{2}}$. From Lemma 2, (i) implies that $error(i, j) \leq \epsilon$ which proves that P' is a ϵ -simplification of P . From lemma 2 again, (ii) implies that $error(i, j + 1) > \frac{\epsilon}{2\sqrt{2}}$. The hypothesis are then similar to the ones of the proof of [2, Theorem 3.4], and a similar reasoning proves the guarantee.

Note that the complexity of Algorithm 1 has not been adressed yet. Indeed, the difficulty lies in the updates of $w(i, j)$ and $b(i, j)$ when a new point p_j is considered. These two variables depend directly on the direction $\overrightarrow{p_i p_j}$. However, when a new point is added, this direction may change drastically: Figure 2 shows an example where the maximal width and maximal backpath are achieved for different vertices of the polyline for $\overrightarrow{p_i p_j}$ and $\overrightarrow{p_i p_{j+1}}$.

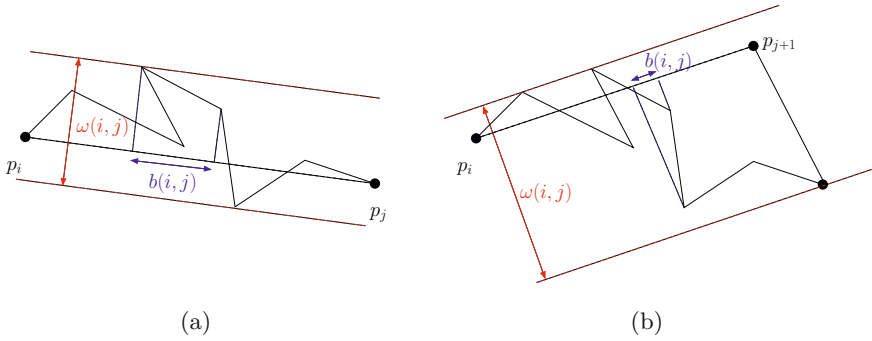


Fig. 2. Updating $\omega(i, j)$ and $b(i, j)$ can be costly in the general case: for instance, the longest backpath is much smaller in (b) for the direction $\overrightarrow{p_i p_{j+1}}$ than in (a) for the direction $\overrightarrow{p_i p_j}$

In the next section, we show how to update efficiently the decisions on $\omega(i, j)$ and $b(i, j)$, with a specification for digital curve to reach a near-linear time complexity.

3 Updating the Approximated Distance Efficiently

In [1], where the approximated distance is defined, an actual computation of the variables $\omega(i, j)$ and $b(i, j)$ is necessary since the problem is to minimize the error. However, as computing the exact values is too expensive, guaranteed approximations are updated when a new point is added. Contrary to [1], in our framework an update of these variables is not necessary, but the decisions “is $\omega(i, j) \leq \frac{\epsilon}{\sqrt{2}}$?” and “is $b(i, j) \leq \frac{\epsilon}{\sqrt{2}}$?” must be exact to ensure the computation of an ϵ -simplification. This section is devoted to the design of new algorithmic approaches to solve these two problems.

3.1 Decision on $\omega(i, j)$

Instead of deciding whether $\omega(i, j)$ is under a given threshold or not, we show that it is enough to check the distance between any point of $P(i, j)$ and the vector $\overrightarrow{p_i p_j}$.

Property 1. Let $d_{max}(i, j) = \max_{p \in P(i, j)} d(p, \overrightarrow{p_i p_j})$. We have $\frac{\omega(i, j)}{2} \leq d_{max}(i, j) \leq \omega(i, j)$.

This property is actually implicitly used in the proof of Lemma 2 in [1, Lemma 4.2]. The authors use the following inequalities to define the approximated distance (the parameters (i, j) are omitted for the sake of lisibility):

$$\max\left(\frac{\omega}{2}, \frac{b}{2}\right) \leq \max(d_{max}, \frac{b}{2}) \leq error \leq \sqrt{2} \max(d_{max}, b) \leq \sqrt{2} \max(\omega, b)$$

In our framework, it is much easier to use d_{max} instead of ω thanks to the algorithm of Chan and Chin [5, Lemmas 1 and 2]. Given an origin point p_i and a set of points $P(i, j)$ we construct the set S_{ij} of straight lines l going through p_i such that $\max_{p \in P(i, j)}(p, l) \leq r$.

To do so, we use the following simple fact: $d(p, l) \leq r \Leftrightarrow$ the straight line l crosses the disk of center p and radius r (see Figure 3(a)). S_{ij} is a cone computed incrementally in $\mathcal{O}(1)$ time considering for a point p_k two new rays defined by p_i and the disk of center p_k and radius r (see Figure 3(b)). As a result, deciding whether $d_{max}(i, j)$ is lower than r or not is equivalent to checking whether the straight line (p_i, p_j) belongs to S_{ij} or not.

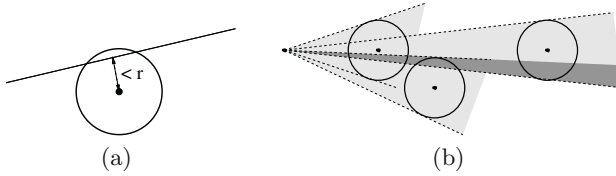


Fig. 3. (a) A line which is at a distance lower than r from a point p crosses the disk of center p and radius r . (b) The cone S_{ij} (dark gray) is computed incrementally as the intersection of the light gray cones.

3.2 Decision on $b(i, j)$

We first show that some particular points, named occulters, play a special role in the decision on $b(i, j)$. Then, we restrict the framework to digital curves to get a better complexity thanks to the following two facts: the computation of the occulters can be mutualized, and the number of occulters is bounded by the error.

General considerations

Definition 4. A *occulter* for the direction \vec{d} is a point p_k such that for all $l < k$, $proj_\alpha(p_k) \gg proj_\alpha(p_l)$. Moreover, an *occulter* is said to be *active* if there is no occulter p'_k with $k' > k$.

Property 2. There is one and only one active occulter for a given direction \vec{d} .

In other words, the active occulter of a curve $P(i, j)$ in the direction \vec{d} is the point equal to $\arg \max(proj_\alpha(p))$. Figure 4 (a) illustrates this definition.

Property 3. The origin of the longest backpath of $P(i, j)$ is an occulter for the direction $\overrightarrow{p_i p_j}$.

Proof. Let $\overrightarrow{p_k p_m}$ be the longest backpath of $P(i, j)$ in the direction $\overrightarrow{p_i p_j}$. Suppose that p_k is not an occulter. Then there exist a point p_l of $P(i, j)$ such that $l < k$ and $proj_\alpha(p_l) \gg proj_\alpha(p_k)$. Thus $\overrightarrow{p_l p_m}$ is a backpath too, and $\|proj_\alpha(\overrightarrow{p_l p_m})\| > \|proj_\alpha(\overrightarrow{p_k p_m})\|$, which is a contradiction with the fact that $\overrightarrow{p_k p_m}$ is the longest backpath.

Property 4. Consider the set of occuters $\{occ_j, j = 1 \dots n\}$ of $P(i, k)$. Let occ_{max} be the active occuter. Consider all the backpaths $\overrightarrow{occ_j p_k}$ ending at p_k , if any. Then $\|proj_\alpha(\overrightarrow{occ_{max} p_k})\| > \|proj_\alpha(\overrightarrow{occ_j p_k})\|$ for all j .

Proof. By definition, $proj_\alpha(occ_{max}) \gg proj_\alpha(occ_j)$ for all j , and the result follows straightforwardly.

Putting together the previous properties, we see that updating the active occuter is the key point of the computation of the longest backpath.

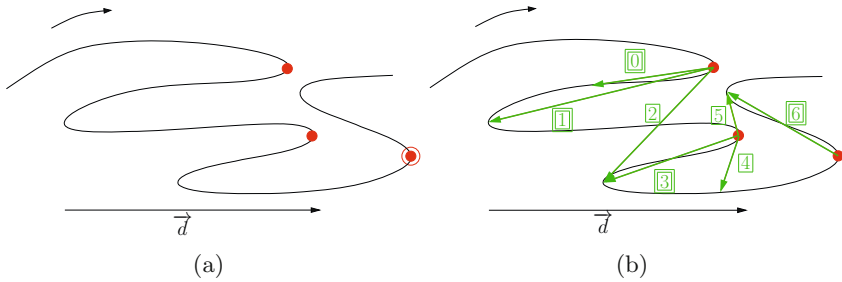


Fig. 4. (a) Occuters for the direction \vec{d} . The only active occuter is circled. (b) Backpaths in the direction \vec{d} : the origin of all the backpaths is an occuter, but only the double-squared backpaths need to be considered in the algorithm.

Suppose that we are computing the backpaths in direction \vec{d} for the curve $P(i, j)$. Suppose that all the points of $P(i, k)$ have been processed, and that the point p_{k+1} is added. If $p_k p_{k+1}$ is a positive shift then nothing needs to be done:

- p_{k+1} cannot be the origin of a backpath since it's the last point processed.
- p_{k+1} is not the end of the longest backpath since $proj_\alpha(p_{k+1}) \gg proj_\alpha(p_k)$.

The case when $p_k p_{k+1}$ is a negative shift is detailed in Algorithm 2. Figure 4(b) illustrates the difference cases of Algorithm 2. Only the double-squared backpaths are considered in the algorithm. Indeed, the backpaths number 2 and 5 are not taken into account because of Property 4. The backpath number 4 is not considered either because the end of the backpath follows a positive shift: we know for sure that the backpath number 3 is longer. However, note that the backpath number 0 is considered: the length of this backpath may be greater than the threshold.

According to Algorithm 1 we see that Algorithm 2 must be applied for any direction $\overrightarrow{p_l p_m}$ where p_l and p_m are any two points of the curve, with $l < m$. In the general case, for a polygonal curve of n points, there are $\mathcal{O}(n^2)$ such directions, that can be computed as a preprocessing of the curve. In the case of a digital curve embedded in an image of size $n \times n$, the possible directions are known *a priori* but $\mathcal{O}(n^2)$ directions must still be considered. However, we see in the following that the computation of the backpaths can be mutualized in the case of digital curves.

Algorithm 2. Active occulter update and backpath computation for a direction \vec{d}

- 1 Let occ_{max} be the active occulter for $P(i, k)$ in direction \vec{d} .
 - 2 **if** $\overrightarrow{p_k p_{k+1}}$ is negative **then**
 - 3 **if** $\overrightarrow{p_{k-1} p_k}$ is positive **then**
 - 4 **if** $proj_\alpha(p_k) \gg proj_\alpha(occ_{max})$ **then**
 - 5 $occ_{max} = p_k$
 - 6 The vector $\overrightarrow{occ_{max} p_{k+1}}$ may be a backpath.
-

Digital curves specificities. In the following, we consider 8-connected digital curves, but the algorithm also works for 4-connected curves.

An elementary shift is a vector $\overrightarrow{p_k p_{k+1}}$. For a digital curve, there are only 8 elementary shifts, given by the chain codes, and denoted $\vec{e}_i, i = 0 \dots 7$ in the following. We also classify the directions $\vec{d} = (d_x, d_y)$ into 8 octants as illustrated in Figure 5

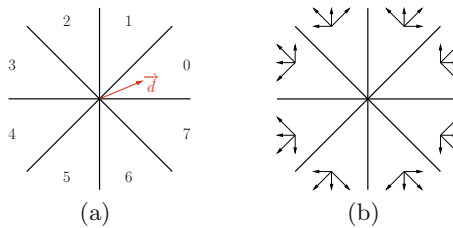


Fig. 5. (a) Clustering of the directions into octants: for instance, the direction \vec{d} belongs to the octant 0. (b) Illustration of the elementary positive shifts for each octant.

Lemma 4. Consider any elementary shift \vec{e}_i . Then, the sign of $\vec{d} \cdot \vec{e}_i$ is the same for all the directions \vec{d} of a given octant.

Proof. An octant clusters all the directions with a fixed sign for d_x and d_y , and such that the order on d_x and d_y is the same. For instance, in the octant 0, we find all the directions such that $0 \leq d_y < d_x$. Since the components $e_{i,x}$ and $e_{i,y}$ of \vec{e}_i lie in $\{-1, 0, 1\}$, the sign of $\vec{d} \cdot \vec{e}_i$ only depends on the signs and order of d_x and d_y , which ends the proof.

As a consequence, for all the directions of a given octant the elementary positive and negative shifts are the same. In Figure 5(b), all the elementary positive shifts are depicted for each octant. Thus, the tests of Algorithm 2 lines 2-3 are the same for all the directions of a given octant. Nevertheless, on line 4, the projections of two points on a direction \vec{d} are compared, and this test is not so easy when an octant of directions is considered. In the following the octant 0 is

considered, but a similar reasoning can be done for the other cases. Consider the Figure 6(a), where the plane is divided into four areas according to the position of a point p :

- for any point q in the gray area, we have $proj_\alpha(q) \gg proj_\alpha(p)$ for any direction of the octant 0;
- for any point q in the dashed area, we have $proj_\alpha(p) \gg proj_\alpha(q)$ for any direction of the octant 0;
- in the white area, the order of the projections changes, as illustrated in Figure 6(b-d).

Thus, we do not have only one active occulter to update anymore, but a set of occulters for each octant. Any active occulter is associated to an interval of angles for which it is actually the active occulter. From Property 2, we derive that these intervals are disjoint and that their union is the interval $[0, \frac{\pi}{4}]$. Algorithm 3 describes how to update the active occulters for the directions of octant 0.

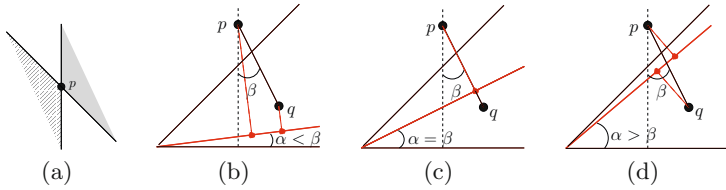


Fig. 6. When the point q in the white area (a), the order of the projections of p and q on the line of angle α changes according to the angle β (b-d)

The complexity of Algorithm 3 depends on the number of active occulters. The following lemma is used in the complexity analysis in Section 4 to prove that the number of occulters per octant is bounded by the approximation error in the case of digital curves.

Lemma 5. *For a digital curve and for a given octant, there is at most one active occulter per line and per column of \mathbb{Z}^2 .*

Idea of the proof. When two points p and q are on the same row or the same column, then for all the directions of a given octant, either p is an occulter for q or q is an occulter for p .

List of forbidden directions. From Algorithm 1, we see that the length of the longest backpath is tested for each point, which defines a new direction. Moreover, we see from Algorithm 2 line 6 that for each negative shift, we can have as many backpaths as active occulters. All in all, testing individually all the possible backpaths when a new point is added is too costly. To solve this problem, we propose to maintain a “set” of the directions for which there exist a backpath of length greater than the error ε .

Algorithm 3. Update of the list of active occulterers for the octant 0

```

1 Let  $p$  be the last point added, we want to check if  $p$  is an active occulter.
2 forall the active occulterers  $p_i(\alpha_{i_{min}}, \alpha_{i_{max}})$  do
3    $v = \overrightarrow{p_i p}$ 
4   if  $\overrightarrow{v} \cdot (1, 0) < 0$  and  $\overrightarrow{v} \cdot (1, 1) < 0$  then
5      $\lfloor$   $p$  is not an active occulter
6   if  $\overrightarrow{v} \cdot (1, 0) > 0$  and  $\overrightarrow{v} \cdot (1, 1) > 0$  then
7      $\lfloor$   $p_i$  is not an active occulter anymore
8      $\lfloor$   $p$  is an active occulter on  $[0, ?]$  with  $\alpha_{i_{min}} < ? \leq \frac{\pi}{4}$ 
9   if  $\overrightarrow{v} \cdot (1, 0) > 0$  and  $\overrightarrow{v} \cdot (1, 1) < 0$  then
10    compute the angle  $\beta$  ; /* see Figure 6 */
11    if  $\alpha_{i_{min}} \leq \beta < \alpha_{i_{max}}$  then
12       $\lfloor$   $p$  is an active occulter on  $[0, \beta]$ 
13       $\lfloor$   $p_i$  is an active occulter on  $[\beta, \alpha_{i_{max}}]$ 
14    if  $\beta < \alpha_{i_{min}}$  then
15       $\lfloor$   $p_i$  is still an active occulter
16    if  $\beta \geq \alpha_{i_{max}}$  then
17       $\lfloor$   $p_i$  is not an active occulter anymore
18       $\lfloor$   $p$  is an active occulter on  $[0, ?]$  with  $\alpha_{i_{max}} \leq ? \leq \frac{\pi}{4}$ 
19 +similar process for symmetrical cases (roles of  $p_i$  and  $p$  inverted)

```

This set actually consists of a list of intervals defined as follows. Consider a backpath $\overrightarrow{p_k p_m}$ of length l . Then the length of the projection of this backpath on the direction \overrightarrow{d} is a function of l and the angle between $\overrightarrow{p_k p_m}$ and \overrightarrow{d} . This is illustrated in the Figure 7 for a given backpath of length l and angle α , and a given error ε , the interval of directions for which the projection of the backpath has a length greater than ε is computed easily. Eventually, such an interval is computed for each backpath of length greater than ε , and the list of all these intervals is called the list of forbidden directions.

At the end, we have the following equivalence: $b(i, j)$ is greater than ε if and only if the direction $\overrightarrow{p_i p_j}$ belongs to the list of forbidden directions. This equivalence enables to test efficiently $b(i, j)$ in $\mathcal{O}(\log(n_i))$ for n_i intervals (see Algorithm 1).

4 Theoretical and Experimental Results

4.1 Complexity and Guarantee Analysis

Theorem 2. Algorithm 1, combined with Algorithm 2 and Algorithm 3, compute in $\mathcal{O}(n \log(n_i))$ an ε -simplification of a digital curve P under the Fréchet distance with at most the number of vertices of an optimal $\frac{\varepsilon}{4\sqrt{2}}$ -simplification.

Proof. The proof of the guarantee is given by Lemma 3. The complexity of the algorithm lies in: (i) the number of active occulterers n_o and (ii) the size of

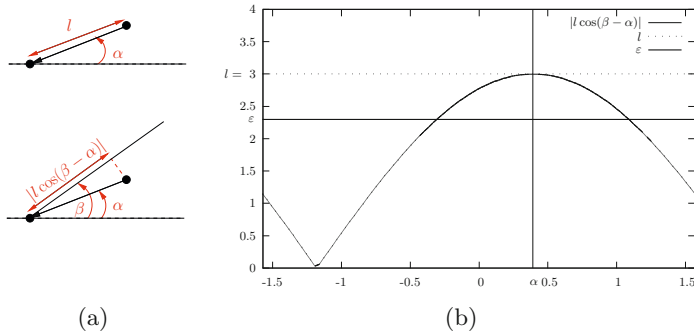


Fig. 7. (a) Illustration of the function defining the length of the projection of a back-path. (b) Plot of this function: when a threshold ϵ is fixed, the interval of angles for which the length is greater than ϵ is defined.

the list of intervals of forbidden directions n_i . Indeed, the general complexity is $\mathcal{O}(n(n_o + \log(n_i)))$. Nevertheless, putting together Lemma 5 and the fact that the width is bounded by the error, we get that n_o is also upper bounded by the error, which ends the proof.

n_i is more difficult to bound since one interval may be added after each negative shift, but we see in the following section that experimentally, the algorithm runs in linear time.

4.2 Experimental Behaviour

Figure 8 illustrates the results of our algorithm for a noisy flower with 5 extremities, for three different values of the ϵ parameter. The images were generated with the Imagen toolkit [9].

In Figure 9(a), runtime results are depicted for noisy synthetic data of increasing size: a circle, a flower with 5 extremities, and a phase accelerating flower with 5 extremities. We see that the general behaviour is clearly linear in time. In

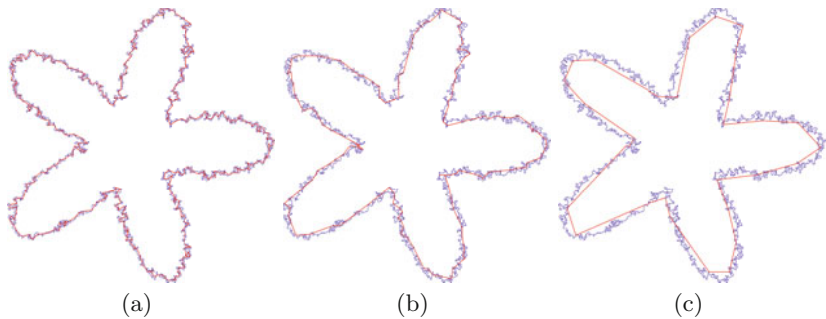


Fig. 8. Results of our algorithm on a noisy flower for a parameter ϵ equal to 3 in (a), 8 in (b) and 15 in (c)

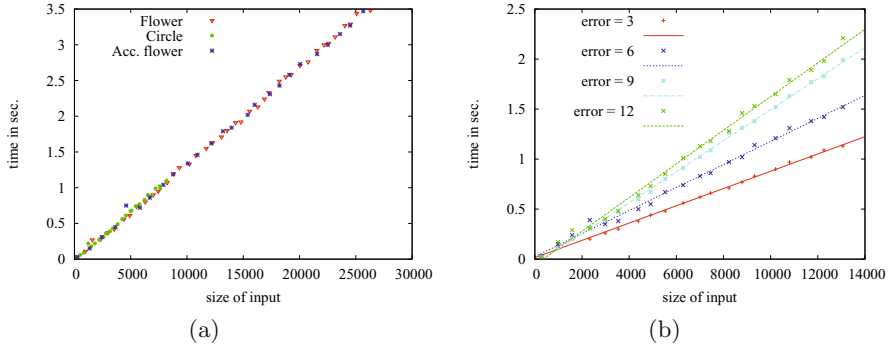


Fig. 9. Runtime results for noisy synthetic shapes. In (b), a noisy flower with five extremities is used.

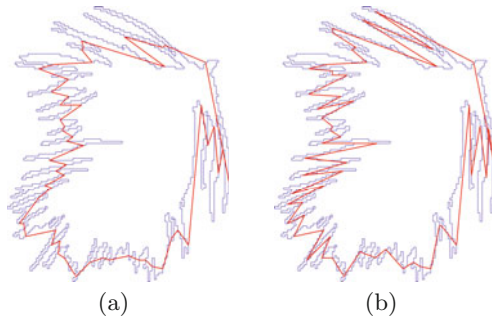


Fig. 10. Comparison of the simplification results on a leaf, with an error equal to 8 with a criterion on the width only in (a), and approximated Frechet distance – width and backpath length – in (b)

Figure 9(b), we study the runtime for different values of ϵ for noisy synthetic flowers of increasing size. Once again, we see a linear behaviour for any value of ϵ . A more detailed study of the evolution of the slope would give some hints to refine the theoretical complexity analysis.

Lastly, in Figure 10 we compare the result of the approximated Frechet simplification algorithm (where the width and the backpaths length are taken into account) with the result of a simplification algorithm with a width criterion only (points are added while $w(i, j)$ is lower than the error). We see that for the same error, even if the polygons returned by the two algorithms roughly have the same number of vertices (56 in (a), 60 in (b)), the sharp features of the shape are better preserved with the approximated Frechet distance.

5 Future Works

The first perspective is to refine the theoretical complexity from $\mathcal{O}(n \log(n))$ to $\mathcal{O}(n)$ since the experimental results show a linear behaviour. Another perspective

is to extend this algorithm (and its complexity) to general polygonal curves. Indeed the main idea of the algorithm is to cluster the directions used for the projection according to the directions of elementary shifts. This clustering is possible if the minimal angle between two elementary shifts directions is known, which is trivial in the case of digital curves, but could be computed as a preprocessing for polygonal curves.

References

1. Abam, M.A., de Berg, M., Hachenberger, P., Zarei, A.: Streaming algorithms for line simplification. In: SoCG 2007, pp. 175–183. ACM, New York (2007)
2. Agarwal, P.K., Har-Peled, S., Mustafa, N.H., Wang, Y.: Near-linear time approximation algorithms for curve simplification. *Algorithmica* 42(3-4), 203–219 (2005)
3. Alt, H., Godau, M.: Computing the Fréchet distance between two polygonal curves. *International Journal of Computational Geometry* 5(1), 75–91 (1995)
4. Buzer, L.: Optimal simplification of polygonal chains for subpixel-accurate rendering. *Comput. Geom. Theory Appl.* 42(1), 45–59 (2009)
5. Chan, W.S., Chin, F.: Approximation of polygonal curves with minimum number of line segments. In: Ibaraki, T., Iwama, K., Yamashita, M., Inagaki, Y., Nishizeki, T. (eds.) ISAAC 1992. LNCS, vol. 650, pp. 378–387. Springer, Heidelberg (1992)
6. Driemel, A., Har-Peled, S., Wenk, C.: Approximating the Fréchet distance for realistic curves in near linear time. In: SoCG 2010, pp. 365–374. ACM, New York (2010)
7. Godau, M.: A natural metric for curves—computing the distance for polygonal chains and approximation algorithms. In: Jantzen, M., Choffrut, C. (eds.) STACS 1991. LNCS, vol. 480, pp. 127–136. Springer, Heidelberg (1991)
8. Imai, H., Iri, M.: Polygonal approximations of a curve: formulations and algorithms. In: *Computational Morphology*, pp. 71–86. Elsevier, Amsterdam (1988)
9. Lachaud, J.O.: ImaGene, <https://gforge.liris.cnrs.fr/projects/imagene/>
10. Pelletier, S.: Computing the Fréchet distance between two polygonal curves, <http://www.cim.mcgill.ca/~stephane/cs507/Project.html>
11. Sriraghavendra, E., Karthik, K., Bhattacharyya, C.: Fréchet distance based approach for searching online handwritten documents. In: ICDAR 2007: Int. Conference on Document Analysis and Recognition, pp. 461–465. IEEE Computer Society, Los Alamitos (2007)

Distance between Separating Circles and Points

Peter Veelaert

University College Ghent, Engineering Sciences - Ghent University Association,
Schoonmeersstraat 52, B9000 Ghent, Belgium
Peter.Veelaert@hogent.be

Abstract. The family of separating circles of two finite sets S_1 and S_2 in the plane consists of all the circles that enclose S_1 but exclude S_2 . We prove that the maximum and minimum distance between a point p and any separating circle in this family can be found by examining only a finite subset of circles, although the family itself is infinite. In addition, we introduce the concept of elementary circular separations to clarify some of the properties of separating circles.

1 Introduction

We consider the problem of finding the shortest and largest distance between a point and a family of separating circles. More precisely, let S^- and S^+ be two sets of planar points. We consider the family of separating circles for which the points of S^- always lie inside the circles and the points of S^+ outside. We want to show that shortest and largest distance between a planar point and any member of this family can be found in a finite number of steps, although the family itself is infinitely large. This result is also of use for finding distances between two families of separating circles, or between a straight line and a family of circles.

O'Rourke et al already noted that there is a unique smallest separating circle, which can be found by convex programming [7]. The largest separating circle cannot be found by convex programming and is not always unique. They also showed that any largest separating circle of finite radius must either pass through 3 points of S^+ , or must pass through 2 points of S^+ and one of S^- . Thus, the largest circle can be found by examining all the 3-point subsets of $S^- \cup S^+$.

Our results on separating circles are also relevant for digitized circles. In the past two distinct views have been developed on digital curves [6]. In one view one regards a digital curve as a set of points for which there is a continuous curve that is *passing close* to these points. This is the preferable viewpoint when regarding a digital curve as the digitization of a continuous curve. A second view is preferred, when one considers a digital curve as the boundary of a digital set. Then the digital set can be *separated* from its complement by a continuous curve. The separation/closeness dichotomy reappears when we examine the basic properties of digital curves. Elemental subsets are the smallest subsets of a curve for which it is meaningful to define closeness. Thus, an *elemental circular subset* has 4 points, since a circle can always pass at zero distance from 3 points. The word elemental comes from robust estimation theory where it refers to subsets that are just large enough to produce an estimate. As for separation, we will introduce the concept of *elementary circular separations*, containing as few as 3 points.

Several authors have proposed algorithms to determine circular separability [3–8]. Invariably, either the parameter domain of the separating circles, or its projection on the plane of circle centers, plays an important role in these algorithms. In this work we shall prove some additional results that link the parameter domains to elementary circular separations.

2 Elementary Circular Separations and Parameter Domains

Elementary circular separations are introduced to provide a means of separating the points of a set $S = S^- \cup S^+$ unambiguously by circles, even when a circle passes through some points of S . Since 3 points uniquely define a circle, the definition of elementary circular separations is based on 3-point subsets, although more than 3 points are allowed when they are circular by coincidence.

Let $R \subseteq S$ be a set of 3 or more points lying on a common circle conveniently denoted by C_R . Let $\sigma_R : R \rightarrow \{+, -\}$ be a function that attributes a sign to each point of R .

Definition 1. *The sign function σ_R is called an elementary circular separation if there exists a circle C not equal to C_R , such that all the points of the preimage $\sigma_R^{-1}(+)$ lie outside C and all the points of the preimage $\sigma_R^{-1}(-)$ lie inside C .*

By definition C does not pass through any of the points of R . We will often use the shorthands $R^+ = \sigma_R^{-1}(+)$, and $R^- = \sigma_R^{-1}(-)$. Note that although the total number of points must be at least 3, one of the subsets R^+ or R^- may be empty.

Since the circle C_R only passes through the points in R and not through any other point of S , the separation function σ_R can be extended to a function σ_S over the entire set in an unambiguous way, where σ_S attributes the minus sign to a point that lies in the interior of C_R , and the plus sign when it is exterior. We will use the shorthands $S^+ = \sigma_S^{-1}(+)$, and $S^- = \sigma_S^{-1}(-)$.

2.1 Path-Connected Parameter Sets

The extension of σ_R from an elementary separation to the entire set is straightforward. We will show that the converse is also true. When a family is defined as the set of circles that separate S into given parts S^+ and S^- , we show that this family can always be represented by an elementary circular separation. To accomplish this we define path-connected sets in the parameter space of separating circles. Let S be a non-empty finite set of points in the plane. A circle is defined by an equation of the form

$$(x - a)^2 + (y - b)^2 - r^2 = 0,$$

which can always be rewritten as

$$x^2 + y^2 - 2ax - 2by + c = 0, \tag{1}$$

with $a^2 + b^2 - r^2 = c$. Conversely, (1) corresponds to a real circle provided $c \leq a^2 + b^2$. We are interested in circles that separate the points of S unambiguously, which is the case if they do not pass through any of the points of S . This leads to a 3D parameter space from which some planes are removed.

Definition 2. Let S be a finite subset of \mathbb{R}^2 and let $U \subset \mathbb{R}^3$ be the set parameter points (a, b, c) such that

$$\begin{aligned} a^2 + b^2 &> c, \\ x_i^2 + y_i^2 - 2ax_i - 2by_i + c &\neq 0, (x_i, y_i) \in S. \end{aligned} \tag{2}$$

The closure of a path-connected subset of U is called a *domain of separating circles*.

The set U is an open set consisting of multiple disconnected subsets, which may be bounded or unbounded. If C is any circle not passing through the points of S , then the unique domain in which its parameters lie will be denoted as $D(C)$.

It is not difficult to see that the path-connected subsets of U are exactly the families of circles that always separate the set S in the same two parts. According to Definition 2 the subset U consists of all the points in \mathbb{R}^3 for which

$$x_i^2 + y_i^2 - 2ax_i - 2by_i + c \neq 0$$

for $(x_i, y_i) \in S$. If we let (a, b, c) vary along a continuous path $(a(t), b(t), c(t))$, $t \in [0, 1]$ then if a point p lies inside the circle defined by $(a(0), b(0), c(0))$ it will also be inside the circle defined by $(a(1), b(1), c(1))$ since a path cannot contain circles that pass through p . Similarly, exterior points will remain exterior along the path. Therefore, the connected sets in U have the form

$$\begin{aligned} a^2 + b^2 &> c, \\ x_i^2 + y_i^2 - 2ax_i - 2by_i + c &> 0, (x_i, y_i) \in \sigma_S^{-1}(+), \\ x_j^2 + y_j^2 - 2ax_j - 2by_j + c &< 0, (x_j, y_j) \in \sigma_S^{-1}(-), \end{aligned} \tag{3}$$

where σ_S is a circular separation of S . Since a domain was defined as the closure of a connected subset of U , a domain is a set that satisfies the following inequalities:

$$\begin{aligned} a^2 + b^2 &\geq c, \\ x_i^2 + y_i^2 - 2ax_i - 2by_i + c &\geq 0, (x_i, y_i) \in \sigma_S^{-1}(+), \\ x_j^2 + y_j^2 - 2ax_j - 2by_j + c &\leq 0, (x_j, y_j) \in \sigma_S^{-1}(-). \end{aligned} \tag{4}$$

We shall denote this domain as $D(\sigma_S)$.

2.2 Polyhedral and Polytopal Domains

It is often assumed that a domain of separating circles is always polyhedral. For a general set S this is not always true, however. We shall make this more precise. The equation $a^2 + b^2 = c$ defines a paraboloid [7, 8]. For each point (x_i, y_i) in S , the equation

$$x_i^2 + y_i^2 - 2ax_i - 2by_i + c = 0$$

represents a plane tangent to this paraboloid. The tangent point is $(a, b, a^2 + b^2)$. Half-spaces of the form $x_i^2 + y_i^2 - 2ax_i - 2by_i + c \geq 0$ correspond to points of S^+ , and are always oriented towards the paraboloid. Halfspaces that correspond to points of S^- are oriented away from the paraboloid.

There is also an unbounded non-polyhedral domain which consists of the circles that do not contain any of the points of S , that is, $S^+ = S$ and $S^- = \emptyset$. This domain is not polyhedral as one of its boundary surfaces is the paraboloid $c = a^2 + b^2$. All other domains are polyhedral, however.

Proposition 1. *Let S be a finite set. Let C be a circle not passing through any point of S , giving rise to a circular separation σ_S . Then $D(C)$ is a H -polyhedron if and only if S^- is non-empty. Furthermore, $D(C)$ is a polytope if $\text{conv}S^+ \cap \text{conv}S^-$ contains a non-empty open set.*

Proof. First, if S^- contains a single point p_1 , then the intersection of $D(C)$ with the surface $c = a^2 + b^2$ contains exactly one point, which represents the circle with radius $r = 0$ centered at p_1 . If S^- contains more than one point, each circle must have a strictly positive radius, and the intersection of $D(C)$ with the paraboloid $c = a^2 + b^2$ is therefore empty. In either case, we can discard the inequality $a^2 + b^2 > c$. Hence, $D(C)$ is a H -polyhedron, that is, the intersection of a finite set of closed half-spaces.

Second, the set defined by (4) is bounded provided there are no parameter points for circles with infinite radius. For this it suffices that S^+ cannot be separated from S^- by a straight line, even if we discard the points on that line. This is equivalent to stating that $\text{conv}S^+ \cap \text{conv}S^-$ contains a non-empty open set. \square

Both conditions of Proposition 1 are satisfied if the interior of the convex hull of S^+ contains at least one of the points of S^- . Therefore, the simplest example of a polytopal domain is when S^+ contains three points p_1, p_2, p_3 , and S^- a fourth point p_4 which lies inside the triangle $p_1p_2p_3$. The polytope is then bounded by four planes. These planes delimit four halfspaces, three oriented towards the paraboloid, and one oriented away from the paraboloid.

2.3 Polytopal Domains and Elementary Circular Separations

There is a direct relation between elementary circular separations and domains. Since an elementary circular separation can be extended unambiguously to a global separation, we define $D(\sigma_R)$ as $D(\sigma_R) = D(\sigma_S)$ where σ_S is the extension of σ_R . Assume furthermore, that the domain $D(\sigma_R)$ is polytopal. Then each **face** of the polytope corresponds to a point of S , i.e., a face lies in a plane of the form

$$x_i^2 + y_i^2 - 2ax_i - 2by_i + c = 0$$

where $(x_i, y_i) \in S$. Similarly, each **vertex** of the polytope corresponds to an elementary circular separation. To be precise, a vertex lies at the intersection of three or more parameter planes of the form

$$\begin{aligned} x_1^2 + y_1^2 - 2ax_1 - 2by_1 + c &= 0, \\ \dots & \\ x_n^2 + y_n^2 - 2ax_n - 2by_n + c &= 0, \end{aligned} \tag{5}$$

where $(x_1, y_1), \dots, (x_n, y_n)$ lie on a common circle. By attributing signs to the points we can establish an elementary circular separation consistent with Definition 1. The relative position of the vertex with respect to the polytope determines the sign of each point. If the polytope is contained in the halfspace $x_1^2 + y_1^2 - 2ax_1 - 2by_1 + c \geq 0$, then (x_1, y_1) receives a positive sign, otherwise (x_1, y_1) receives a negative sign.

An **edge** of the polytope always corresponds to a family of circles that have two points in common, as we can verify by explicit calculation. To be precise, the edge

that connects a vertex (a_1, b_1, c_1) with a vertex (a_2, b_2, c_2) , corresponds to a family of circles of the form

$$x^2 + y^2 - 2(a_1(1 - t) + a_2t)x - 2(b_1(1 - t) + b_2t)y + c_1(1 - t) + c_2t$$

where $0 \leq t \leq 1$. Since all coincidence relations are preserved by an affine transformation, without loss of generality, we may assume that the vertices are $(a_1, b_1, c_1) = (-1, 0, c_1)$ and $(a_2, b_2, c_2) = (1, 0, c_2)$. The above expression then simplifies to

$$c_1(1 - t) + c_2t - 2(-1 + 2t)x + x^2 + y^2. \tag{6}$$

By letting t take two distinct values t_1 and t_2 , we find that the intersection points are $(1/4(-c_1 + c_2), \pm 1/4\sqrt{-(c_1 - c_2)^2 - 8(c_1 + c_2)})$. Because these coordinates are independent of t_1 and t_2 , each circle in (6) passes through the same two points.

Fig. 1 shows an example. The elementary circular separation with $R^- = \{(1, 1), (2, 1)\}$, and $R^+ = \{(1, 0), (2, 0)\}$, induces the signs shown in Fig. 1(a). The gray points belong to S^- and lie inside the separating circles, the black points belong to S^+ . There are infinitely many circles that separate S into S^+ and S^- . Fig. 1(a) shows one of these circles. Fig. 1(b) shows the polytopal domain, which has 5 vertices. The gray area in Fig. 1(c) shows the possible positions of the circle centers, which can be found by projecting the domain onto the ab -plane. Each vertex of the domain corresponds to one circle, also shown in Fig. 1(c). Each circle, and therefore each vertex, determines an elementary circular separation.

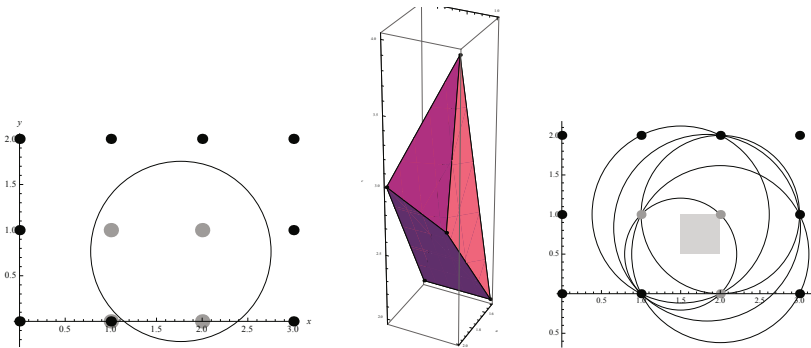


Fig. 1. Family of separating circles

Fig. 2 shows a family of circles whose parameters lie on a polytope edge, all passing through two common points. Clearly, if a planar point p lies inside one of these circles, then p lies in at least one of the two circles that correspond to the vertices of the edge. In other words, the area covered by all the members of the family is the same as the area covered by the two circles of the vertices. This property can be generalized. Let $C_{(a,b,c)}$ denote the circle with parameters (a, b, c) .

Proposition 2. *Let σ_S be a circular separation with polytopal domain $D(\sigma_S)$. If a planar point p lies inside one of the circles with parameters in $D(\sigma_S)$, then there is a vertex $v = (a, b, c)$ of $D(\sigma_S)$ such that p lies inside $C_{(a,b,c)}$.*

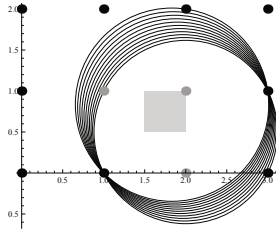


Fig. 2. Circles that correspond to a polytope edge. All the circles pass through two common points.

Proof. The polytope $D(\sigma_S)$ contains all the circles that separate S into two sets S^- and S^+ . The half-space H of circles containing the point $p = (x_0, y_0)$ is determined by

$$x_0^2 + y_0^2 - 2ax_0 - 2by_0 + c < 0.$$

The point p lies inside one of the circles of $D(\sigma_S)$ provided $H \cap D(\sigma_S) \neq \emptyset$. If $H \cap D(\sigma_S)$ is non-empty, it forms a polytope which contains all the circles that contain p . Since $D(\sigma_S)$ is a polytope, $H \cap D(\sigma_S)$ contains at least one vertex v of $D(\sigma_S)$. Hence, the circle $C_{(a,b,c)}$ contains p . \square

Thus the area covered by the interior of the circles in $D(\sigma_S)$ is the same as the area covered by the interior of the circles corresponding to the vertices of $D(\sigma_S)$.

3 Properties of Elementary Circular Separations

We prove two additional properties of elementary circular separations. First, determining whether a circular set and a sign map form an elementary circular separation is straightforward.

Proposition 3. *Let $\sigma_R : R \rightarrow \{+, -\}$ be a map that attributes signs to the points of a circular set R . Then the map σ_R is an elementary circular separation if and only if $\sigma_R^{-1}(+)$ can be separated by a straight line from $\sigma_R^{-1}(-)$.*

Proof. We use the shorthands $R^+ = \sigma_R^{-1}(+)$, and $R^- = \sigma_R^{-1}(-)$. First we show that the preimages R^+ , R^- of an elementary circular separation σ_R can always be separated by a straight line. Let D be the common circle on which the points lie. Let L be any straight line separating R^+ from R^- . Then L divides the open disk bounded by D into two parts, L^+ and L^- , with $R^+ \subset L^+$ and $R^- \subset L^-$. Furthermore, L crosses D in two distinct points v and w . Let q denote any point that lies on the bisector of v and w and in L^+ . Then the circle C passing through v , q , w satisfies the conditions required by Definition [1](#) that is, all the points in R^+ lie outside C and all the points in R^- lie inside C .

Conversely, suppose the points of $R^+ \cup R^-$ lie on a common circle, but that R^+ cannot be linearly separated from R^- . Since they cannot be linearly separated, the

intersection of their convex hulls, i.e., $\text{conv}R^+ \cap \text{conv}R^-$, is non-empty. However, since the points lie on a circle, none of the points in R^+ lies in $\text{conv}R^-$, and vice-versa. It follows that there are two points p_1, p_2 in R^+ and two points p_3, p_4 in R^- such that the line segment p_1p_2 crosses the line segment p_3p_4 . If there exists a circle C that separates R^+ from R^- , then C would also separate p_1, p_2 from p_3, p_4 . Therefore it is sufficient to prove that even these 4 points cannot be separated by a circle, with p_1, p_2 outside the circle and p_3, p_4 inside the circle.

This part of the proof is illustrated in Fig. 3, which shows four points on a common circle D , and the bisectors of the pairs $\{p_1, p_3\}$, $\{p_3, p_2\}$, $\{p_2, p_4\}$, and $\{p_4, p_1\}$. These bisectors pass through the circle center and they divide the plane into 4 open disjoint segments S_1, \dots, S_4 . The bisector of $\{p_1, p_3\}$ divides the plane into two open half-planes. If a circle C has to satisfy Definition 1, then p_1 must lie outside C and p_3 inside C , and the center of C must lie in the half-plane which contains p_3 . Furthermore, since p_2 must also lie outside C , it follows that the center of C must lie in the sector S_3 . On the other hand, by considering p_1, p_2 , and p_4 we find that the center must also lie in the sector S_4 . Since $S_3 \cap S_4 = \emptyset$, this is impossible. \square

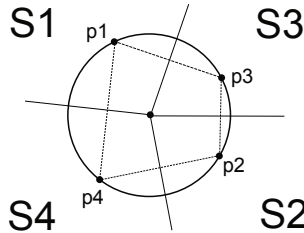


Fig. 3. Illustration of the proof of Proposition 3

With Proposition 3 we can list all elementary circular separation functions that can be defined on a finite circular set. If we divide the points of a circular set into linearly separable parts, these parts always belong to two disjoint circle segments. Hence, each elementary circular separation corresponds to a partitioning of a circular set into two contiguous parts.

O’Rourke et al showed that any largest separating circle of finite radius must either pass through 3 points of S^+ , or it must pass through 2 points of S^+ and one of S^- [7]. As a related result, we will show that in a domain there is at least one circle passing through 2 points of S^+ and one point of S^- .

Proposition 4. *Let σ_S be a circular separation with polytopal domain $D(\sigma_S)$. Then $D(\sigma_S)$ always contains a vertex defined by an elementary circular separation σ_R with $|\sigma_R^{-1}(+)| \geq 2$ and $|\sigma_R^{-1}(-)| \geq 1$.*

Proof. We use the shorthands $S^+ = \sigma_S^{-1}(+)$ and $S^- = \sigma_S^{-1}(-)$. Let C be any circle in the domain $D(\sigma_S)$. We will deform C continuously until its parameters coincide with an elementary circular separation that satisfies the above requirements. We will start by deforming C until it touches two points p_1^+, p_2^+ of S^+ . Then we will deform it further until it touches a third point p_3^- that lies in S^- .

Since the domain is bounded, C separates S into two subsets S^+ and S^- that cannot be linearly separated. Clearly, this is only possible if $|S^-| \geq 1$, and $|S^+| \geq 3$. First, we increase the radius of C until the circle passes through a point p_1 of S^+ , as shown in Fig. 4(a). This is always possible since $|S^+| \geq 3$.

Second, consider the tangent to C passing through the point p_1 . Since S^+ and S^- cannot be separated linearly, there must be at least one other point of S^+ that lies on same side of the tangent as all the points of S^- . Hence, by moving the center a of C along the line p_1a away from p_1 , we can further increase the radius of the circle until it passes through p_1 and a second point p_2 of S^+ . Fig. 4(b) illustrates this.

Finally, we arbitrarily select a point q of $|S^-|$ that does not lie on the line p_1p_2 . We distinguish two cases.

Case A. The point q lies at the same side from the line p_1p_2 as the center of C . Then we start to move the center along the bisector of p_1 and p_2 in the direction of p_1p_2 until the circle passes through p_1 , p_2 and q or one of the other points of S^- . This is illustrated in Fig. 4(c). Note that it may be necessary to move the center to the other side of p_1, p_2 .

Case B. The chosen point q lies at the side from the line p_1p_2 which is the opposite of the side of the center of C , as shown in Fig. 4(d). In this case we increase the radius of the circle by moving its center a along the bisector of p_1 and p_2 away from the line p_1p_2 until C passes through p_1, p_2 and q or one of the other points of S^- .

In both cases, however, the motion of the center of the circle may stop when the circle touches a point of S^+ before it reaches a point of S^- . First, consider how this can happen in case A. Fig. 4(e) shows that the circle has hit a third point p_3 of S^+ before it reaches q . Note that this point must always lie at the side of p_1p_2 that is opposite to q . If this happens, it suffices to replace the chord p_1p_2 , by one of the smaller chords, for example p_1p_3 , and continue the shrinking process along the bisector of the new chord. If we then hit another point of S^+ we can keep replacing the chords by smaller chords. Since S^+ is finite this process must end until we hit one of the points in S^- . Second, we consider how this can happen in case B. Fig. 4(f) shows that the circle has hit a point p_3 , which must lie at side of p_1p_2 that is opposite to q . Clearly, q lies at the same side of the line p_2p_3 as the center of the circle. That is, we can replace the chord p_1p_2 by the chord p_2p_3 and continue as in case A.

In either case, we can deform the circle until it passes through two points p_1 and p_2 of S^+ and one point of S^- . □

In the example shown in Fig. 1, three of the 5 vertices satisfy the conditions of Proposition 4. There is also a vertex with $|R^+| = 3, |R^-| = 0$. One of these four vertices yields the largest separating circle.

4 Distances and Geometric Properties

Once the domains of the families of separating circles have been calculated several geometric properties can be verified immediately, since they directly relate to the topological relations between the polytopal domains. For example, given two families of circles there exists a pair of concentric circles, one from each family, provided the circle centers of the two families intersect. This section considers a more involved computation:

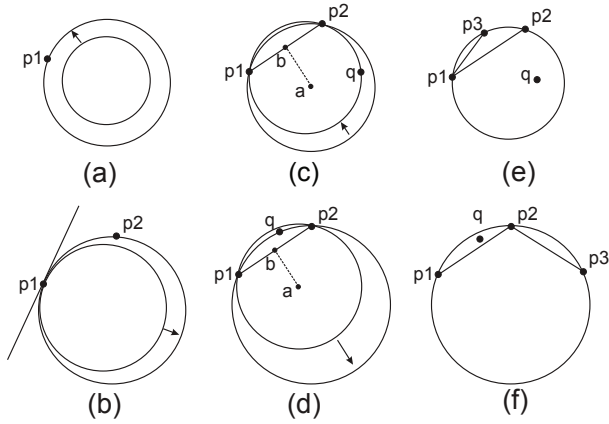


Fig. 4. Illustration of the proof of Proposition 4

the distance between a family of circles and a point. The distance between a point p and a circle C is defined as the shortest distance between p and any point q on C , and will be denoted as $d(p, C)$. If $D(\sigma_S)$ is a polytopal domain of circles, then we define the maximum distance between a point p as $\max_{(a,b,c) \in D(\sigma_S)} d(p, C_{(a,b,c)})$. That is, as maximum of the distances between p and any circle in $D(\sigma_S)$. Likewise, the minimum distance is defined as $\min_{(a,b,c) \in D(\sigma_S)} d(p, C_{(a,b,c)})$. We will prove that the maximum distance can be found by examining only the circles that correspond to the vertices of a domain. We start with a simplified case.

Lemma 1. *Let $V = \{(a_1, b_1, c_1), (a_2, b_2, c_2)\}$ be a set of two parameter points, and let P denote their convex span. Let p be any point in the plane. Then there is a point $u \in C_{(a_1,b_1,c_1)} \cup C_{(a_2,b_2,c_2)}$ such that*

$$\max_{(a,b,c) \in P} d(C_{(a,b,c)}, p) = d(u, p).$$

Furthermore, if p does not lie on one of the circles of P , then there is a point $w \in C_{(a_1,b_1,c_1)} \cup C_{(a_2,b_2,c_2)}$ such that $\min_{(a,b,c) \in P} d(C_{(a,b,c)}, p) = d(w, p)$.

Proof. The proof follows from simple geometric considerations, illustrated in Fig. 5. Fig. 5(a) shows two circles, one centered around c_1 and one around c_2 , and an ellipse passing through q_1 and q_2 . This ellipse is part of the locus of points that are equidistant to both circles. This locus also comprises a hyperbola (not shown here). The ellipse consists of those equidistant points that lie inside one circle but outside the other circle.

The family of circles with parameters in P consists of circles that pass through the points q_1 and q_2 , and can be represented as

$$x^2 + y^2 - 2(a_1(1 - t) + a_2t)x - 2(b_1(1 - t) + b_2t)y + c_1(1 - t) + c_2t \quad (7)$$

with $0 \leq t \leq 1$. With q_1, q_2, c_1, c_2 we construct 5 straight line segments. Together with the ellipse they divide the plane into 8 distinct regions, shown in Fig. 5(b). For example, region R_7 is bounded by the ellipse, and the lines c_2q_1 and c_2q_2 .

The maximum distance between a planar point p and the family of circles (7) depends on the region in which p lies. When p lies either in R_3 or R_4 the maximum distance between p and (7) is the distance between p and q_1 . For example, when p lies in R_4 , this maximum is attained by a circle whose center lies on the straight line passing through p and q_1 . In that case, the circle lies "behind" the point q_1 as seen from p . Similarly, when p lies either in R_1 or R_2 , the maximum distance is equal to distance between p and q_2 . When p lies in either R_6 or R_8 , the maximum distance between p and (7) is the same as the distance between p and the circle centered around c_1 . In fact, this is the furthest a circle in this family can move away from p . Finally, when p lies in either R_5 or R_7 , the maximum distance is equal to the distance between p and the circle centered around c_2 .

In all the cases considered the maximum distance is given by either the distance to the circle in (7) for which $t = 0$, that is $C_{(a_1, b_1, c_1)}$, or for which $t = 1$, that is $C_{(a_2, b_2, c_2)}$, or by the distance to one of the two points q_1 or q_2 which are common to both circles.

For the minimum distance, the proof proceeds in a similar way. We only give a brief sketch. Fig. 5(c) illustrates the proof when the polytope consists of a line segment between the two parameter points (a_1, b_1, c_1) and (a_2, b_2, c_2) . Fig. 5(c) shows how the two circles, and their equidistant ellipse and equidistant hyperbola divide the plane in 8 regions. When p either lies in R_2, R_3, R_6 or R_7 there is a circle of the form (7) passing through p . This case is excluded from the theorem. If p does not lie on a circle of P , then p either lies in R_1, R_4, R_5 or R_8 . When p lies in R_1 or R_5 , the closest circle is $C_{(a_1, b_1, c_1)}$. If p lies in R_4 or R_8 , the closest circle is $C_{(a_2, b_2, c_2)}$. Hence, the minimum distance is either zero, or equal to $\min(d(p, C_{(a_1, b_1, c_1)}), d(p, C_{(a_2, b_2, c_2)}))$. □

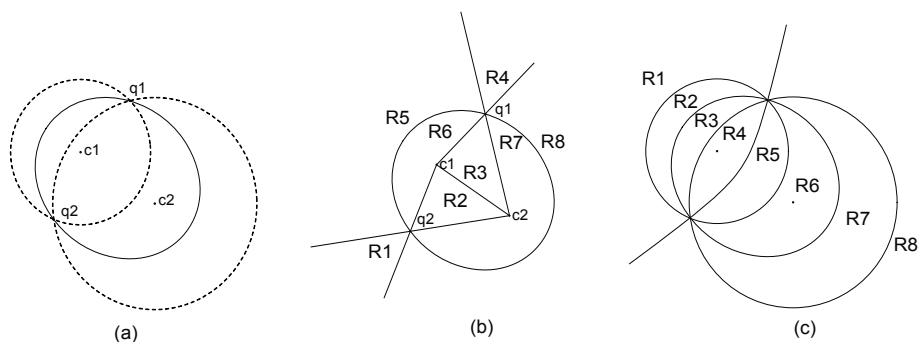


Fig. 5. (a) Two circles shown with dashed lines, and an ellipse of points that are equidistant to both circles. (b) The ellipse and straight lines through the centers and the common points divide the plane into 8 regions.

We now extend this result to a more general setting.

Theorem 1. *Let σ_S be a circular separation with polytopal domain $D(\sigma_S)$. Let V be set of vertices of $D(\sigma_S)$, and let W denote the set $\bigcup_{(a,b,c) \in V} C_{a,b,c}$. Let p be any point in the plane. Then there is a point u in W such that $\max_{(a,b,c) \in D(\sigma_S)} d(C_{(a,b,c)}, p) = d(u, p)$. Furthermore, if p does not lie on one of the circles of $D(\sigma_S)$, then there is a point w in W such that $\min_{(a,b,c) \in D(\sigma_S)} d(C_{(a,b,c)}, p) = d(w, p)$.*

Proof. We use the shorthand $S^+ = \sigma_S^{-1}(+)$ and $S^- = \sigma_S^{-1}(-)$. It suffices to show that the point u always lies on one of the circles $C_{(a,b,c)}$ where $(a, b, c) \in V$. Let $(a_0, b_0, c_0) \in D(\sigma_S)$ be the parameters of a circle for which we have $d(C_{(a_0,b_0,c_0)}, p) = \max_{(a,b,c) \in P} d(C_{(a,b,c)}, p)$. We will show that we can always replace $C_{(a_0,b_0,c_0)}$ by a circle $C_{(a,b,c)}$, with $(a, b, c) \in V$ whose distance to p is at least as large as that of $C_{(a_0,b_0,c_0)}$. Showing that $(a, b, c) \in V$ is equivalent to showing that $C_{(a,b,c)}$ contains at least three points of $S^+ \cup S^-$. We start by assuming that $C_{(a_0,b_0,c_0)}$ does not contain any point of $S^+ \cup S^-$. Then we can always move the center (a_0, b_0) away from p along the line passing through (a_0, b_0) and p . If the radius is kept constant the distance will increase. Since this is impossible, $C_{(a_0,b_0,c_0)}$ contains at least one point of $S^+ \cup S^-$.

Next, we assume that $C_{(a_0,b_0,c_0)}$ contains just one point q_1 of $S^+ \cup S^-$. Let L denote the line passing through (a_0, b_0) and p . First, suppose q_1 lies on L . Then it is clear that we can move the center along L while keeping the distance to p fixed until we hit a second point of $S^+ \cup S^-$. Second, suppose q_1 does not lie on L . Let q'_1 denote the mirror image of q_1 by the reflection across L . We define a pencil of circles of the form $\textcircled{7}$ with common points q_1, q'_1 and center on L . Let q_3 be the first point of $S^+ \cup S^-$ that we hit when we move in this pencil (a_0, b_0) away from p , and let q_4 be the first point of $S^+ \cup S^-$ hit when we move (a_0, b_0) towards from p . Since P is bounded both points must exist. Let $C_{(a_1,b_1,c_1)}$ be the circle passing through q_1, q'_1, q_3 and let $C_{(a_2,b_2,c_2)}$ denote the circle passing through q_1, q'_1, q_4 . According to Lemma \square the maximal distance between p and the pencil of circles is either the distance to $C_{(a_1,b_1,c_1)}$, to $C_{(a_2,b_2,c_2)}$ or to one of the points q_1, q'_1 . However, q_1, q'_1 lie at the same distance of p , and q_1 is a point of $S^+ \cup S^-$. Hence, we can replace $C_{(a_0,b_0,c_0)}$ always by a circle that contains two points q_1, q_2 of $S^+ \cup S^-$, and state that the maximal distance is either the distance to this circle or one of the points of $S^+ \cup S^-$.

Finally, we can repeat the previous argument, but now for the pencil of circles defined by the common points q_1, q_2 , until we hit a third point of $S^+ \cup S^-$. The maximal distance is either the distance to this new circle or to one of the points q_1, q_2 , which are both in $S^+ \cup S^-$. For minimum distances, the proof is similar. \square

The above result also shows how an algorithm can compute the correct maximum or minimum distance. First, for each edge of the polytope we compute the maximum distance between p and the circles of the edge. This can be done by examining the position of p with respect to the straight lines shown in Fig. $\textcircled{5}$ where the circles correspond to the vertices adjacent to the edge. If p lies in one of the regions R_1, \dots, R_4 the maximum distance is either $d(p, q_1)$ or $d(p, q_2)$. Otherwise, it is the distance to one of the two circles corresponding to the vertices. After this distance has been computed for all the edges, it suffices to take the global maximum.

To find the minimum distance one first has to verify whether $p = (x_p, y_p)$ does not lie on one the circles in P . It suffices to verify whether the plane

$$x_p^2 + y_p^2 - 2ax_p - 2by_p + c = 0$$

crosses the polytope P . If p does not lie on any of the circles, the distance can be found by computing the distance for each circle corresponding to a vertex, and by taking the global minimum.

5 Time Complexity and Conclusion

Clearly, the computation of the distance between a point and the separating circles depends mainly on the computation of the complete face lattice (faces, edges, and vertices) of the parameter polytope. Clarkson and Shor give a randomized algorithm in 3D with expected time complexity of $O(n \log n)$, where n is the number of half-spaces. This algorithm was derandomized by Chazelle [1, 2]. The polytope of separating circles for $|S| = n$ can therefore be computed in $O(n \log n)$ time, where $O(n \log n)$ is an upper bound for the expected number of edges, as well as vertices in the face lattice. The computation of the maximal and minimal distances requires the examination of all edges and vertices, where each vertex or edge can be processed in constant time. Hence distances can be computed in $O(n \log n)$ time.

References

1. Chazelle, B.: An optimal convex hull algorithm in any fixed dimension. *Discrete and Computational Geometry* 10, 377–409 (1993)
2. Clarkson, K.L., Shor, P.: Applications of random sampling in computational geometry, II. *Discrete Computational Geometry* 4, 387–421 (1989)
3. Coeurjolly, D., Gerard, Y., Revelles, J.-P., Tougne, L.: An elementary algorithm for digital arc segmentation. *Discrete Applied Mathematics* 139, 31–50 (2004)
4. Damaschke, P.: The linear time recognition of digital arcs. *Pattern Recognition Letters* 16, 543–548 (1995)
5. Fisk, S.: Separating points sets by circles, and the recognition of digital disks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8, 554–556 (1986)
6. Kim, C.E., Anderson, T.A.: Digital disks and a digital compactness measure. In: *Annual ACM Symposium on Theory of Computing*, pp. 117–124 (1984)
7. O’Rourke, J., Koraraju, S.R., Meggido, N.: Computing circular separability. *Discrete and Combinatorial Geometry* 1, 105–113 (1986)
8. Roussillon, T., Tougne, L., Sivignon, I.: On three constrained versions of the digital circular arc recognition problem. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 34–45. Springer, Heidelberg (2009)

Optimal Consensus Set for Annulus Fitting

Rita Zrour, Gaëlle Largeteau-Skapin, and Eric Andres

Laboratory XLIM, SIC Department,
University of Poitiers BP 30179, UMR CNRS 6712
86962 Futuroscope Chasseneuil Cedex, France

Abstract. An annulus is defined as a set of points contained between two circles. This paper presents a method for fitting an annulus to a given set of points in a 2D images in the presence of noise by maximizing the number of inliers, namely the consensus set, while fixing the thickness. We present a deterministic algorithm that searches the optimal solution(s) within a time complexity of $O(N^4)$, N being the number of points.

Keywords: Digital geometry; shape fitting; consensus set; outliers; digital arc; annulus.

1 Introduction

Detection of basic geometric properties such as lines, planes and circles are essential tasks in the field of image analysis and computer vision. The main objective of this paper is annulus fitting to a given set of points. For instance, annulus fitting is useful for shape approximation [4] and image segmentation [6]. In this paper, we present a novel method that, given an arbitrary 2D point cloud, finds annuli of a given width that minimizes the number of outliers, or alternatively maximizes the number of inliers. One of the possible application of our method in the digital space is the fitting of discrete analytical circle of Andres [2]. The set of points which do fit the model is also called consensus set. The idea of using such consensus sets was proposed for the RANdom Sample Consensus (RANSAC) method [9], which is one of the most widely used in the field of computer vision. However RANSAC is inherently probabilistic in its approach and does not guarantee any optimality while our method is both deterministic and optimal in the size of the consensus set.

Different algorithms detecting annuli have been proposed. Most of these algorithms minimize the thickness of the annuli (digital circle). Among them, some consider that no noise is present in the image, and concentrate only on the problem of recognition instead of the fitting problem [16,5,11]. Some algorithms deal with outliers [7,10,11] but the number of outliers is usually predefined [10,11] and the problem consists in minimizing the width.

One frequently used approach in annulus recognition stems from the O'Rourke et al. [13] disk recognition method that transformed it into a problem of circular separability; they use a mapping that raises every point (x, y) to the paraboloid

$z = x^2 + y^2$. Using this mapping, every circle in the primal space corresponds to a plane cutting the paraboloid in the dual space and thus circular separability is transformed to a plane separability in the dual space. This separability problem is then solved in linear time using Megiddo’s or Dyer linear algorithms [8][12]. Recently, in the discrete geometry community, a simple online linear-time algorithm for recognition of digital circular arcs has been proposed in [15] based on the idea of O’Rourke et al. [13]; this algorithm can be used in an incremental way, with a complexity $O(n^{4/3})$ whenever a new point is added by using an optimization proposed in [3]. Using O’Rourke’s mapping, an annulus of fixed area in the primal space, corresponds to two parallel planes of fixed vertical thickness in the dual space. The vertical thickness of the two parallel planes corresponding to an annulus is however not fixed when the thickness w of the annuli and not the area of the annuli is fixed, as is the case of our fitting problem. In this case, the dual approach does not seem to simplify the problem.

To our knowledge there is no work neither in the continuous domain nor in the discrete domain that treated the problem of optimal annulus fitting with fixed width. Our main contribution is that we find exact solution(s) by minimizing the number of outliers.

The rest of the paper is organized as follows: in section 2 we expose the problem of annulus fitting, present some properties of annuli and explain how we can find all the consensus sets. In sections 3 we show how to build an annulus from three points. Section 4 provides the algorithm for finding the optimal annulus and shows some results. Finally Section 5 states some conclusions and perspectives.

2 Annulus Fitting

An annulus A of width ω and radius R centered at $C(C_x, C_y)$, is defined by the set of points in R^2 satisfying two inequalities:

$$S = \{(P_x, P_y) \in \mathbb{R}^2 : R^2 \leq (P_x - C_x)^2 + (P_y - C_y)^2 \leq (R + \omega)^2\} \quad (1)$$

where $C(C_x, C_y) \in \mathbb{R}^2$ and $R, \omega \in \mathbb{R}_+$.

Using the above annulus model, our fitting problem is then described as follows: given a finite set $S = \{(P_x, P_y) \in \mathbb{R}^2\}$ of n points such that $n \geq 3$, and given a width ω we would like to find an annulus A of width ω such that it contains the maximum number of points in S . Points $(P_x, P_y) \in S \cap A$ are called inliers; otherwise they are called outliers. It should be noted that $n \geq 3$ since if S contains less than 3 points, the fitting problem has an infinity number of solutions.

We denote B_i (resp. B_e) that we call the internal (resp. external) border of the annulus, the set of points located at distance R (resp. $R + \omega$) from C .

2.1 Annuli and Their Consensus Sets

Our approach is focused on inlier sets, also called consensus sets. Since S is finite, it is obvious that the number of different consensus sets for the annulus

is finite as well. Thus, if we can find all different consensus sets \mathcal{C} from a given set S , we just need to verify the size of each \mathcal{C} and find the maximum one as the optimal solution. Then the following question comes up naturally: is it possible, given a width ω to find all the consensus sets of S ? if the answer is positive, how can we do it? In the section [2.2](#), we will answer both questions by giving some properties related to annuli.

2.2 Annular Characterizations

The following theorem states that given a width ω , and given an annulus A covering a set of points S , there exists at least another annulus A' of same width, that covers S and passes through at least 3 points of S .

Theorem 1. *Let S be a set of n ($n \geq 3$) points in \mathbb{R}^2 . Let $A = (C(C_x, C_y), R, \omega)$ be the annulus of center $C(C_x, C_y)$, of internal radius R and of width ω such that $\forall (P_x, P_y) \in S, R^2 \leq (P_x - C_x)^2 + (P_y - C_y)^2 \leq (R + \omega)^2$. Then it exists $A' = (C'(C'_x, C'_y), R', \omega)$ such that:*

$$\exists P_0, P_1, P_2 \in S, \forall i \in [0, 2], P_i \in B_i \text{ or } P_i \in B_e$$

Proof. Let S be a set of n ($n \geq 3$) points in \mathbb{R}^2 . Let $A = (C(C_x, C_y), R, \omega)$ be the annulus of center $C(C_x, C_y)$ with internal radius R and width ω such that A covers S : i.e. $\forall (P_x, P_y) \in S, R^2 < (P_x - C_x)^2 + (P_y - C_y)^2 < (R + \omega)^2$. Our first assumption is that no point of S is on the annulus borders.

The theorem proof is given in three steps: First we show that we can always decrease the radius so as to put at least one point of S on the annulus external borders. We note this point P_0 . In the second step, we show that a rotation centered on P_0 allows to put another point P_1 of S on one border. The last step is more delicate since it consists in changing both the radius and the center position so that a third point P_2 is now on one of the borders of the annulus. Of course, if in any of these steps we put more than one point on the borders than we just skip one or more steps.

- A First step: the radius decreasing. Let R_0 be the distance between the farthest point P_0 of S from the center C of A . The annulus $A_0 = (C(C_x, C_y), (R_0 - \omega), \omega)$ is such that P_0 is on its external border B_e .
- B Second step: the rotation centered on P_0 . Since the rotation center is on the external border (see [A]), we can continuously rotate A_0 to reach a second point P_1 of S .

We note P_1 the first point reached. Let $Rot_{P_0, \theta}$ the rotation centered on P_0 with angle θ , such that θ is the smallest angle verifying:

$$\exists \theta \in [0, 2\pi], \exists P_1 \in S, R^2 \leq (P_{1_x} - C_x)^2 + (P_{1_y} - C_y)^2 \leq (R + \omega)^2$$

We denote $A_1 = (Rot_{P_0, \theta}(C), R, \omega)$.

C Third step: the radius variation. After the first two steps, we obtain an annulus A_1 that has two points P_0 and P_1 of S on its borders. Two configurations can appear: either both points are on the border B_e , or the points are each on a different border (case iii).

i. Both points P_0 and P_1 are on B_e . In this case, any modification on the radius R involves a move of the center along the bisector of segment P_0 and P_1 . When the center C passes exactly between both points, some problems may occur and we have to separate two cases:

* $d(P_0P_1) \geq 2\omega$: in this case, the center can pass between the points and there always exists an annulus of width ω passing through both points. All the points of S can be reached this way and we necessarily encounter a third point of S . In Fig. 1a, we can see the annulus with the two extreme cases, when the radius is $\pm\infty$. Fig. 1b, shows an initial set of points with an initial annulus of center C passing by two external border points P_0 and P_1 (colored black), and the rotated annulus (colored red) by moving the center toward $-\infty$ from C to C_2 in order to reach a third points P_2 . Fig. 1c shows the case when the center moves toward $+\infty$ in order to reach a third points P_2 . The movement of the center between $\pm\infty$ allows to reach a third point.

* $d(P_0P_1) < 2\omega$: when the points P_0, P_1 are on B_e , the circle having the two points on B_e must have at least a radius of ω since if the radius is less than ω , no internal circle and so no annulus can be built. However when $d(P_0P_1) < 2\omega$, the movement of the center from C to C_2 to reach a third points P_3 cannot be done in all the cases. For example in Fig. 1d, $d(P_0P_1) = \omega$ and in this case, the center cannot move along the green line since $\frac{\omega^2}{4} + b^2 > \omega^2$, i.e. b must be greater than $\frac{\sqrt{3}\omega}{2}$. Therefore there exist some points of S that cannot be reached. Fig. 1e shows a case where no annulus having three border points can be constructed from the annulus having P_0 and P_1 on B_e . When this is the case, we have to change the configuration: we choose as point P_2 the point closest to the limit and we build the annulus of width ω with internal circle passing through P_0, P_1 and P_2 (Fig. 1f).

ii. Both points are on different borders.

In Fig. 2a, b, c, there are three possible configurations of two points on different border of the annulus A . In each of these configurations, modifying the radius allows to reach almost all the initial set. If a point P_2 of S is inside the green circle Fig. 2d, e, inside the annulus, it has to be reached by one of the annulus border. However, there exists an area that is not reached by such a variation (in dark in Fig. 2d, e).

If the points of S are all in this dark part, we have to change our strategy: we choose as point P_2 of S the closest to one of the extreme lines and we build the annulus of width ω such that the three points P_0, P_1 and P_2 are on B_e (see Fig. 2g, h). This annulus thus covers the whole dark area where all the other points of S are and it passes through 3 points of S .

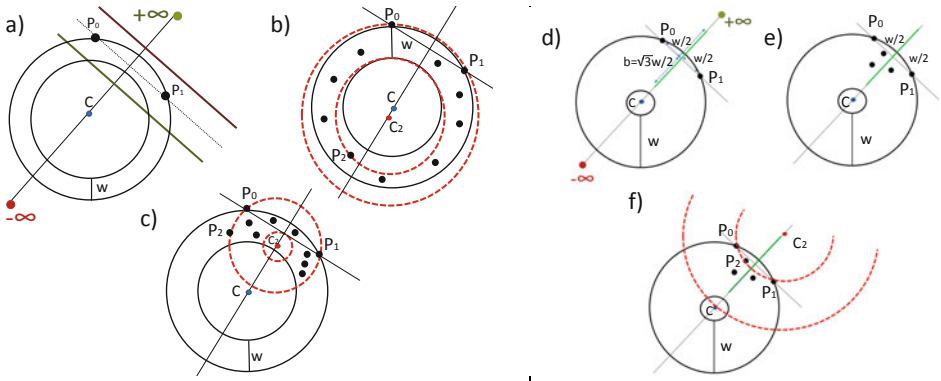


Fig. 1. Third step: a), b) and c), shows case ii where both points P_0 and P_1 are on the external border of A and the distance is greater than 2ω . The center C colored blue must be moved along the line $-\infty$ in b) and $+\infty$ in c) in order to reach a third point P_2 . The new center is C_2 and is colored in red, d), e), f) shows case ii where both points P_0 and P_1 are on the external border of A and the distance between them is less than 2ω . The configuration must be changed by choosing a point P_2 closest to the limit and the new annulus is the one colored in red.

In Fig. 2c, f, where both points are at distance ω ; we have to perform a rotation centered on one of both points until reaching another point P' on the border. If we rotate using P_0 , the point P_1 in Fig. 2f no longer belongs to the border of the annulus. In this case the configuration can be viewed as the same configurations we already addressed: either P' and P_0 are on the internal border or P' is on the external border but not at the distance ω from P_0 (otherwise it would have been reached by a circle centered on P_0 with radius ω passing through P_1 and thus we would already have found the annulus we were searching for). Both cases leads to a third point.

In all cases, if an annulus of width ω covers S , then it is possible to build an annulus of same width that passes through 3 points of S . □

3 Building an Annulus of Width ω from Three Points

The following theorem states that we can build a finite number of annuli of width ω from 3 points (see Fig. 3).

Theorem 2. *There are at most 8 annuli of a given width ω passing through 3 given points P_1, P_2 and P_3 of S .*

Proof.

- If the 3 points are on the same circle: depending on the radius of the circle one or two annuli can be build. If the radius of the circle is greater than ω

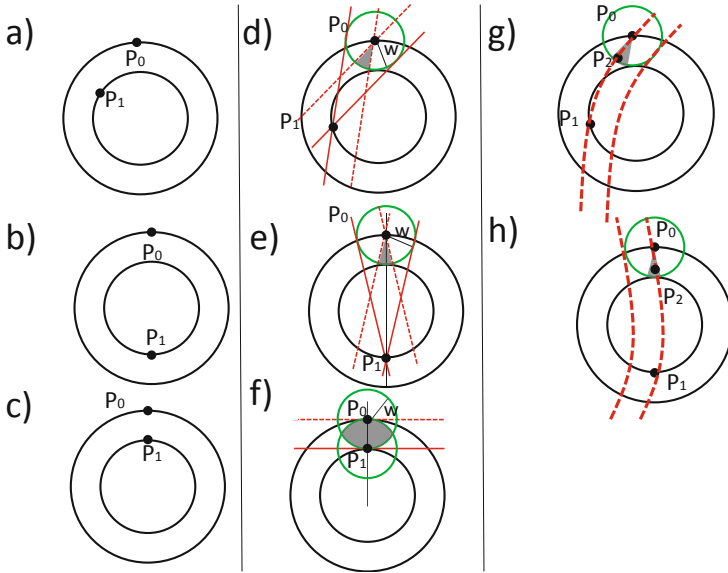


Fig. 2. Third step: case iii, both P_0 and P_1 are on different border of A

then 2 annular are built, the first having the points on the internal border and the other having the points on its external border. When the radius is less than ω then only one annulus having the three points on its internal border can be built.

- If 2 of the 3 points are on one border and the third one on the other: three possible configurations exists either (P_1, P_2 or P_3 is alone on one border) and for each of them we can build at most two annuli (the lonely point is either on B_i or on B_e). The principle of the building process is to build the two circles C' and C'' that passes through the 2 points on the same border and tangent (from the outside or inside) to the circle c of radius ω centered on the third point. (see [14] for more details about the construction).

Fig. 3 shows the different cases of building the annulus, we assume that P_0 and P_2 are on one border and P_1 on the other. Fig. 3a shows the construction method; the circle C of radius ω centered on P_2 is drawn (blue circle), then the two circles C' and C'' passing through P_0 and P_2 and tangent to C are drawn (red circles). Fig. 3b presents the particular case where P_0 is exactly at distance ω from P_1 (i.e. on the circle of center P_1 and radius ω). In this case there is a unique circle that passes through P_0 and P_2 and tangent to C . Finally, there is a particular case when both P_0 and P_2 are inside the circle C (Fig. 3c), in this case it should be stated than no annulus having P_0 and P_2 on one border and P_1 on the other can be found.

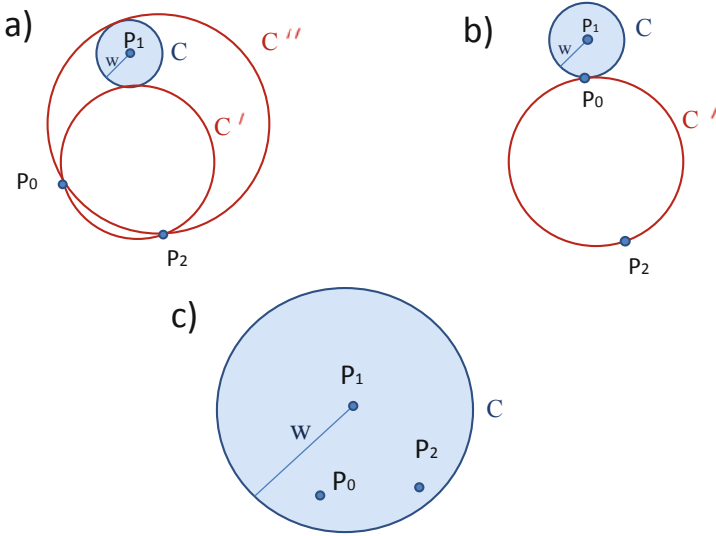


Fig. 3. Building of an annulus of width ω from 3 points: a) P_0 and P_2 are outside the circle C of radius ω centered on P_1 , b) P_0 is on the circle C , c) P_0 and P_2 are inside the circle C

4 Finding the Optimal Fitting Annulus of Width ω

The principle of this naïve algorithm is to test all the possible triplets of points in S . For each triplet, we compute all the possible annuli (see theorem 2) and count the number of inliers. The optimal annulus (or the annuli) is the one that encloses the maximum number of inliers.

4.1 Algorithm

We now present Algorithm 1. Input is a set S of N discrete points and a thickness w of our digital circle model. Output is a set \mathbf{V} of parameter values $(C_x^{op}, C_y^{op}, R^{op})$ corresponding to the fitted annuli that give the optimal consensus sets. The time complexity of the algorithm is $O(N^4)$ because we have N points in S and every combination of three points defines each of the 8 circles, so taking the points three by three has a complexity of $O(N^3)$, then checking how many inliers and outliers we have for a given annulus is done in $O(N)$ time.

4.2 Experiments

We applied our method for 2D noisy digital Andres circles as shown in Fig. 4, 6, 7. For each of these set of points, an annulus of width $\omega = 3$, $\omega = 1$ and $\omega = 1$ is used respectively. Table 1 shows the number of points, the optimal consensus set size as well as the center position and the radius R of the inner circle obtained

Algorithm 1. Annuli fitting while fixing the thickness

```

input : A set  $S = \{P_i\}_{i \in [1, n]}$  of  $N$  points and a thickness  $\omega$ 
output: A list  $\mathbf{V}$  of parameter values  $[(C_x^{op}, C_y^{op}, R^{op}), \dots]$  of the best fitted
annuli  $A$ 

1 begin
2   initialize  $Max = 0$ ;
3   foreach  $i \in [1, n - 2]$  do
4     foreach  $j \in [i + 1, n - 1]$  do
5       foreach  $k \in [j + 1, n]$  do
6          $L = \text{list of the existing annuli passing through } P_i, P_j, P_k$ ;
7         foreach element of  $L$  do
8           initialize  $nb\_inliers = 0$ ;
9           for  $m = 1, \dots, N$  do
10            if  $dist(C, P_m) \geq R$  and  $dist(C, P_m) \leq R + \omega$  then
11               $nb\_inliers = nb\_inliers + 1$ ;
12            if  $nb\_inliers > Max$  then
13              clear( $\mathbf{V}$ );
14               $Max = nb\_inliers$ ;
15              set  $C_x^{op} = C_x, C_y^{op} = C_y, R^{op} = R$ ;
16              put  $(C_x^{op}, C_y^{op}, R^{op})$  in  $\mathbf{V}$ ;
17            else if  $nb\_inliers = Max$  then
18              set  $C_x^{op} = C_x, C_y^{op} = C_y, R^{op} = R$ ;
19              append  $(C_x^{op}, C_y^{op}, R^{op})$  in  $\mathbf{V}$ ;
20   return  $\mathbf{V}$ ;
21 end

```

after the fitting. It should be noted that in Fig. 7, two optimal consensus sets are possible, since two annulus having the same number of inliers can be fitted (two circles in Fig. 7). This proves that our method is capable of detecting all optimal consensus sets. Our method is also applied to an Andres arc of width 1 as shown in Fig. 5, we can see that the arc of width $\omega = 1$ is detected.

5 Conclusion and Perspectives

In this paper we have presented a new method for fitting annulus to a set of points while fixing the width of the annulus. Our approach is costly in terms of computation time however, its main advantage is that it guarantees optimal result from the point of view of maximal consensus set: we are guaranteed to fit an annulus with the least amount of outliers. This is the first time, to the author's best knowledge that an exact optimal methods dealing with the problem of annulus fitting with a fixed width with outliers has been proposed. One of the future work concerns the complexity of our approach. Our approach is rather brute-force and we obtain a $O(N^4)$ complexity. The question of improving this

Table 1. The number of points and the optimal consensus set size for each of Fig. 4, 5, 6, 7

Figures	Number of points	Center position	R	Thickness w	Opt. consensus set size
Fig. 4	289	(31,31)	13	3	286
Fig. 5	121	(101.581,102.226)	86	1	118
Fig. 6	119	(31,31)	14	1	65
Fig. 7	309	(31,31) (49,49)	19	1	114

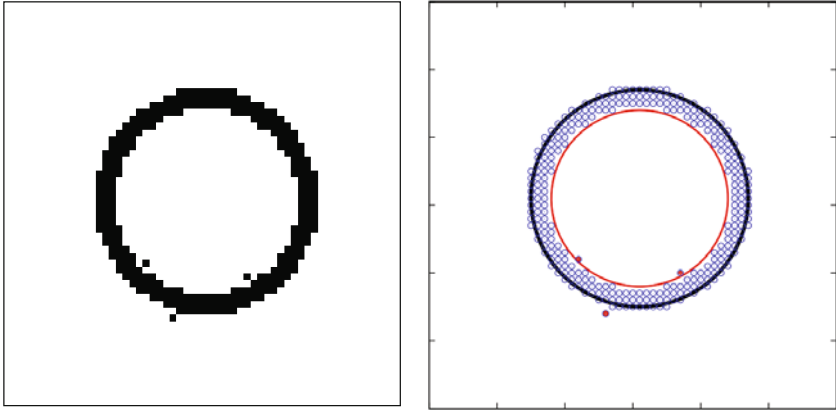


Fig. 4. Annulus fitting for a noisy digital Andres circle of width 3

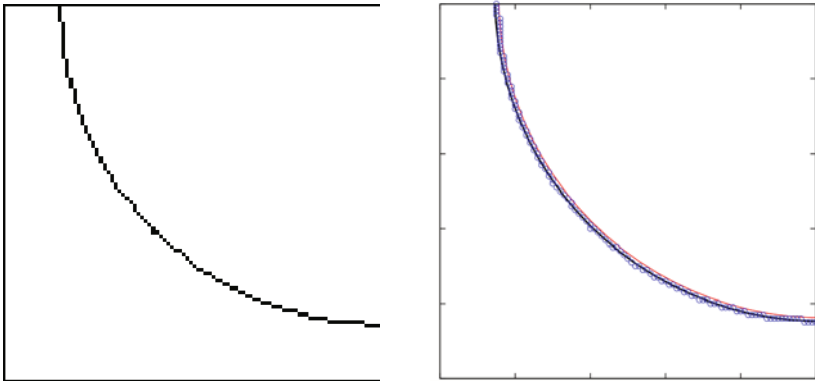


Fig. 5. Annulus fitting for a noisy digital Andres arc of width 1

complexity is open. We have wondered if this complexity may not be optimal. Indeed, it should be noted that fitting a digital plane (corresponding to two parallel continuous planes) to a given set of points in the presence of noise by maximizing the number of inliers is solved in $O(N^3 \log N)$ [17]. As we have

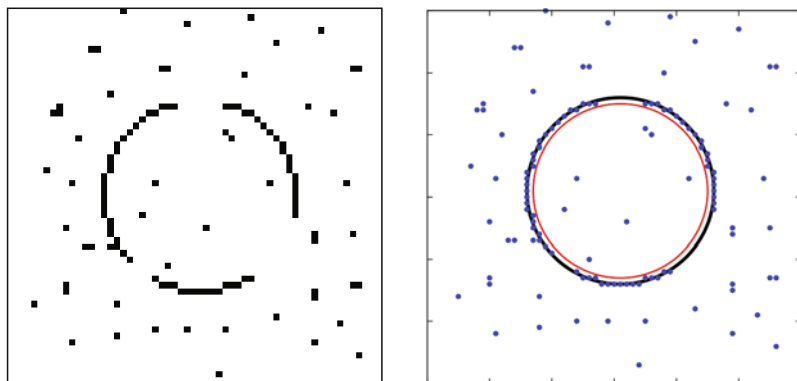


Fig. 6. Annulus fitting for a noisy image of a digital Andres circle of width 1

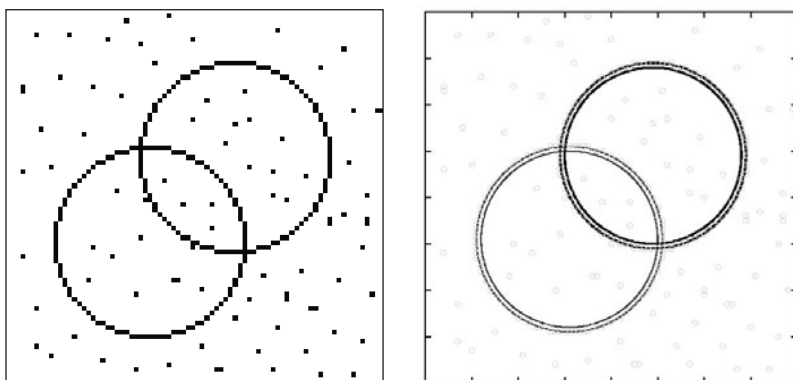


Fig. 7. Annulus fitting for two noisy digital Andres circle of width 1

explained in the introduction, there does not seem to be an immediate way of transforming our problem into a plane fitting problem with a O'Rourke type mapping [13]. However, if such a way exists, we would still face a $O(N^3 \log N)$ lower complexity limit. One planned extension is also the fitting of 3D annuli.

References

1. Agarwal, P.K., Har-Peled, S., Varadarajan, K.R.: Approximating extent measures of points. *Journal of the ACM* 51(4), 606–635 (2004)
2. Andres, E., Jacob, M.A.: The discrete analytical hyperspheres. *IEEE Transactions on Visualization and Computer Graphics* 3, 75–86 (1997)
3. Coeurjolly, D., Gérard, Y., Reveillès, J.P., Tougne, L.: An elementary algorithm for digital arc segmentation. *Discrete Applied Mathematics* 139(1-3), 31–50 (2004)

4. Da Fontoura Da Costa, L., Cesar Jr., R.M.: Shape analysis and classification: Theory and practice, 1st edn. CRC Press, Inc., Boca Raton (2000)
5. De Berg, M., Bose, P., Bremner, D., Ramaswami, S., Ramaswami, G.: Computing constrained minimum-width annuli of point sets. In: Rau-Chaplin, A., Dehne, F., Sack, J.-R., Tamassia, R. (eds.) WADS 1997. LNCS, vol. 1272, pp. 392–401. Springer, Heidelberg (1997)
6. Ding, D., Ping, X., Hu, M., Wang, D.: Range image segmentation based on randomized hough transform. *Pattern Recognition Letters* 26(13), 2033–2041 (2005)
7. Dunagan, J., Vempala, S.: Optimal outlier removal in high-dimensional spaces. *Journal of Computer and System Sciences* 68(2), 335–373 (2004)
8. Dyer, M.E.: Linear time algorithms for two- and three-variable linear programs. *SIAM Journal on Computing* 13(1), 31–45 (1984)
9. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24, 381–395 (1981)
10. Har-Peled, S., Wang, Y.: Shape fitting with outliers. *SIAM Journal on Computing* 33(2), 269–285 (2004)
11. Matousek, J.: On enclosing k points by a circle. *Information Processing Letters* 53(4), 217–221 (1995)
12. Megiddo, N.: Linear-time algorithms for linear programming in r^3 and related problems. *SIAM Journal on Computing* 12(4), 759–776 (1983)
13. O’Rourke, J., Kosaraju, S.R., Megiddo, N.: Computing circular separability. *Discrete and Computational Geometry* 1, 105–113 (1986)
14. Rouche, E., de Comberousse, C., Poincare, H.: *Traité de géométrie*. Edition Jacques Gabay (1900), tomes I et II, 7e édition, Reprint (1997)
15. Roussillon, T., Sivignon, S., Tougne, L.: On three constrained versions of the digital circular arc recognition problem. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 34–45. Springer, Heidelberg (2009)
16. Smid, M., Janardan, R.: On the width and roundness of a set of points in the plane. *International Journal of Computational Geometry* 9(1), 97–108 (1999)
17. Zrour, R., Kenmochi, K., Talbot, H., Buzer, L., Hamam, Y., Shimizu, I., Sugimoto, A.: Optimal consensus set for digital plane fitting. In: *Proceedings of 3DIM 2009 ICCV Satellite Workshop*, pp. 1817–1824 (2009)

Bounds on the Difference between Reconstructions in Binary Tomography

K. Joost Batenburg^{1,2}, Wagner Fortes^{1,3}, Lajos Hajdu^{4,5}, and Robert Tijdeman³

¹ Centrum Wiskunde & Informatica, Amsterdam, The Netherlands
joost.batenburg@cwi.nl

² Vision Lab, University of Antwerp, Belgium

³ Mathematical Institute, Leiden University, The Netherlands

⁴ Institute of Mathematics, University of Debrecen, Hungary

⁵ Number Theory Research Group of the Hungarian Academy of Sciences, Debrecen, Hungary

Abstract. Tomography is concerned with the reconstruction of images from their projections. In this paper, we consider the reconstruction problem for a class of tomography problems, where the images are restricted to binary grey levels. For any given set of projections, we derive an upper bound on the difference between any two binary images having these projections, and a bound on the difference between a *particular* binary image and any binary image having the given projections. Both bounds are evaluated experimentally for different geometrical settings, based on simulated projection data for a range of images.

1 Introduction

The field of *tomography* studies the problem of reconstructing an object from its projections, recorded along a range of viewing angles. Projection images are typically acquired by sending a certain beam (e.g., X-rays) through the object, while measuring the attenuated beam profile that results after beam-object interaction. If a sufficient number of high-quality projection images are available, an accurate reconstruction of the object can be computed using a tomographic reconstruction algorithm [6].

In various applications of tomography, only few projections can be acquired, or the range of available projection directions is limited. Such reconstruction problems are known as *limited-data problems*. In electron tomography, for example, the shape of the sample holder limits the angular range of the projections [10]. In industrial tomography for quality assurance, limitations on the duration of a scan impose an upper bound on the number of projections. Applying classical reconstruction algorithms such as Filtered Backprojection to limited-data problems often results in inferior reconstruction quality. Several approaches have been proposed for overcoming these problems, by incorporating various forms of *prior knowledge* about the object in the reconstruction algorithm. The field of *discrete tomography* focuses on the reconstruction of images that consist of a small, discrete set of grey values [7,8]. By exploiting the knowledge of these grey values in the reconstruction algorithm, it is often possible to compute accurate reconstructions from far fewer projections than required by classical "continuous" tomography algorithms. Still, despite the discrete grey level assumption, the reconstruction problem may be underdetermined and a large number of solutions may exist. In such

cases, it can be important to know how different these solutions can be. If one can give a bound on the maximum difference between two solutions, this also bounds the maximum difference between the unknown ground truth image, of which projections have been measured, and any other solution.

In this paper, we focus on *binary* reconstruction problems, where the unknown image is known to have only two grey levels, 0 and 1. The problem of bounding the difference between binary reconstructions is closely related to the *stability problem*, which concerns the question how the reconstruction changes if the projections are slightly modified [13,12,15]. In [13,14], Van Dalen presented sharp bounds on the difference between any two binary images having the *same* projections for the case of two projections: horizontal and vertical.

The bound we present here is much more general, as it can incorporate an arbitrary number of projections and can even be applied in different geometrical settings (lattice images, as well as discretized continuous images). It is based on the observation first made by Hajdu and Tijdeman in [5], that the Euclidean norm of all binary solutions does not depend on the particular solution, and can be determined directly from the projections. Using Pythagoras' theorem, this allows for bounding the Euclidean norm of the difference between two such solutions. Although the resulting bound will not be strict in general, it is the first general bound for binary tomography problems that can be used for an arbitrary number of projections and for several projection models.

This paper is structured as follows. In Section 2, we define a general class of reconstruction problems and discuss two specific examples of such problems, for lattice images and a strip projection model, respectively. We also introduce notation and recall some basic properties. In Section 3, a general bound is derived on the difference between two binary images having a given set of projections. In Section 4, a bound is derived on the difference between a particular binary image, which is relatively easy to compute, and any binary solution of the tomography problem. Section 5 presents a series of simulation experiments and their results. From these results, the practical value of the proposed bounds can be evaluated for different types of images. Section 6 concludes the paper.

2 Notation and Model

Throughout the discrete tomography literature, several imaging models have been considered. In the *grid model*, an image is formed by assigning a value to each point in a regular grid. In the case of binary images, each point is assigned a value of either 0 or 1; see Chapter 1 of [7]. Here, we consider square grids of the form $A = \{(i, j) \in \mathbb{Z}^2 : 1 \leq i, j \leq c\}$ for $c \geq 1$; see Fig. 1(a). A *projection* is formed by considering the set of parallel lines through one or more grid points in a certain direction $(a, b) \in \mathbb{Z}^2$, with $a \geq 0$ and (a, b) coprime, and summing the values of the points on each line. The grid model can be used to model nanocrystals, consisting of atoms on a grid [9].

In many tomography applications, a *continuous* representation of the object is more realistic, as there is no intrinsic grid structure. In such cases, the unknown image is typically approximated by an image defined on a discrete pixel grid. A common model for computing the projections of such a pixelized image is the *strip model*. In the strip

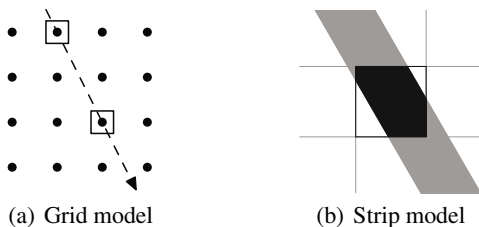


Fig. 1. Two different projection models

model, a projection is computed by considering a set of parallel strips in a given direction and computing a weighted sum of all the pixels that intersect with each strip. The weight is determined by the intersection area between the strip and the pixel [16].

We now define some general notation. An *image* is represented by a vector $\mathbf{x} = (x_i) \in \mathbb{R}^n$. We refer to the entries of \mathbf{x} as *pixels*, although they can also correspond with grid points, in the grid model. The derivation of our main results does not depend on the particular projection model. Throughout this paper we assume that all images are square, consisting of c rows and c columns, where $n = c^2$. A *binary image* corresponds with a vector $\bar{\mathbf{x}} \in \{0, 1\}^n$.

For a given set of k projection directions, the *projection map* maps an image \mathbf{x} to a vector $\mathbf{p} \in \mathbb{R}^m$ of *projection data*, where m denotes the total number of line measurements. As the projection map is a linear transformation, it can be represented by a matrix $\mathbf{W} = (w_{ij}) \in \mathbb{R}^{m \times n}$, called the *projection matrix*. Entry w_{ij} represents the weight of the contribution of x_j to projected line i . Note that for the grid model the projection matrix is a binary matrix, while for the strip model its entries are real values in $[0, 1]$. The projection matrix \mathbf{W} and vector \mathbf{p} can be decomposed into k blocks as

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}^1 \\ \vdots \\ \mathbf{W}^k \end{pmatrix}, \quad \mathbf{p} = \begin{pmatrix} \mathbf{p}^1 \\ \vdots \\ \mathbf{p}^k \end{pmatrix}, \tag{1}$$

where each block \mathbf{W}^d ($d = 1, \dots, k$) represents the projection map for a single direction and each block \mathbf{p}^d represents the corresponding projection data.

From this point on, we assume that the projection matrix has the property that $\sum_{i=1}^m w_{ij} = k$ for all $j = 1, \dots, n$. This property is certainly satisfied for the grid model, as every x_j is counted with weight 1 for exactly one line in each projection direction. The property is also satisfied for the strip projection model, as the total pixel weight for each projection angle is equal to the area of a pixel, which is 1.

The *general reconstruction problem* consists of finding a solution of the system $\mathbf{W}\mathbf{x} = \mathbf{p}$ for given projection data \mathbf{p} , i.e., to find an image that has the given projections. In *binary tomography*, one seeks a binary solution of the system. For a given projection matrix \mathbf{W} and given projection data \mathbf{p} , let $S_{\mathbf{W}}(\mathbf{p}) := \{\mathbf{x} \in \mathbb{R}^n : \mathbf{W}\mathbf{x} = \mathbf{p}\}$, the set of all real-valued solutions corresponding with the projection data, and let $\bar{S}_{\mathbf{W}}(\mathbf{p}) := S_{\mathbf{W}}(\mathbf{p}) \cap \{0, 1\}^n$, the set of *binary solutions* of the system. As the main goal of incorporating prior knowledge of the binary grey levels in the reconstruction is to reduce the

number of required projections, we focus on the case where m is small with respect to n , such that the real-valued reconstruction problem is severely underdetermined.

As the projection matrix is typically not a square matrix, and also does not have full rank, it does not have an inverse. Denote the *Moore-Penrose pseudo inverse* of \mathbf{W} by \mathbf{W}^\dagger (see [4]) and let $\mathbf{x}^* = \mathbf{W}^\dagger \mathbf{p}$. Then \mathbf{x}^* has the following properties (see Chapter 3 of [4]): (i) it is the minimal Euclidean norm solution of the system $\mathbf{W}\mathbf{x} = \mathbf{p}$. (ii) it is orthogonal to the nullspace $\mathcal{N}(\mathbf{W})$ of \mathbf{W} . We call \mathbf{x}^* the *central reconstruction* of \mathbf{p} . The central reconstruction plays an important role in the binary reconstruction problem. We will show in the next section that all binary solutions of the system have equal distance to \mathbf{x}^* , so that one can consider the central reconstruction as lying "in the middle" of all binary solutions.

As our bounds on the distance between binary solutions depend on \mathbf{x}^* , computing \mathbf{x}^* is necessary to compute the corresponding distance bounds. Due to the size of the matrix \mathbf{W} , explicit calculation of \mathbf{W}^\dagger is usually unpractical for large images. As an alternative, an iterative Krylov method for solving the system $\mathbf{W}\mathbf{x} = \mathbf{p}$, called *CGLS* (Conjugate Gradient Least Squares), can be used [11], which is efficient in terms of memory requirements and convergence speed. Apart from numerical errors, applying CGLS to the system $\mathbf{W}\mathbf{x} = \mathbf{p}$ results, after convergence, in the computation of $\mathbf{W}^\dagger \mathbf{p}$, while not computing the matrix \mathbf{W}^\dagger explicitly (see also [12]).

3 A Bound on the Difference between All Binary Solutions

We start this section by showing that the Euclidean norm of any binary solution of $\mathbf{W}\mathbf{x} = \mathbf{p}$ is completely determined by \mathbf{p} .

Lemma 1. *Let $\bar{\mathbf{x}} \in \bar{S}_{\mathbf{W}}(\mathbf{p})$. Then, $\|\bar{\mathbf{x}}\|_2^2 = \frac{\|\mathbf{p}\|_1}{k}$.*

Proof. By the definition of the ℓ_1 -norm, $\|\mathbf{p}\|_1 = \sum_{i=1}^m |p_i| = \sum_{i=1}^m p_i$, since $p_i \geq 0$ ($i = 1, \dots, m$). Also,

$$\sum_{i=1}^m p_i = \sum_{i=1}^m \left(\sum_{j=1}^n w_{ij} \bar{x}_j \right) = \sum_{j=1}^n \left(\sum_{i=1}^m w_{ij} \right) \bar{x}_j = \sum_{j=1}^n k \bar{x}_j, \tag{2}$$

and therefore $\|\mathbf{p}\|_1 = k \sum_{j=1}^n \bar{x}_j$.

As $\bar{\mathbf{x}} \in \{0, 1\}^n$, we have $\|\bar{\mathbf{x}}\|_2^2 = \|\bar{\mathbf{x}}\|_1 = \sum_{j=1}^n \bar{x}_j = \frac{\|\mathbf{p}\|_1}{k}$. □

The following lemma illustrates the importance of the central reconstruction, the shortest real-valued solution in $S_{\mathbf{W}}(\mathbf{p})$, by showing that the binary solutions are the shortest among all integer solutions of the system.

Lemma 2. *Let $\bar{\mathbf{x}} \in \bar{S}_{\mathbf{W}}(\mathbf{p})$ and $\mathbf{y} \in S_{\mathbf{W}}(\mathbf{p}) \cap \mathbb{Z}^n$. Then $\|\bar{\mathbf{x}}\|_2 \leq \|\mathbf{y}\|_2$, with equality if and only if $\mathbf{y} \in \bar{S}_{\mathbf{W}}(\mathbf{p})$.*

Proof. Note that the statement is proved in [5], see Problem 2 and the subsequent paragraph. However, for the convenience of the reader we give the proof here.

We have

$$\|\bar{\mathbf{x}}\|_2^2 = \sum_{i=1}^n \bar{x}_i^2 = \sum_{i=1}^n \bar{x}_i = \sum_{i=1}^n y_i = \frac{\sum_{j=1}^m P_j}{k}. \tag{3}$$

Observing that

$$\sum_{i=1}^n y_i \leq \sum_{i=1}^n y_i^2 = \|\mathbf{y}\|_2^2, \tag{4}$$

with equality if and only if \mathbf{y} is binary, yields the result. □

Supposing the existence of at least two different binary solutions, Lemma 1 allows us to derive an upper bound for the Euclidean distance between those solutions.

Theorem 1. *Let $\bar{\mathbf{x}}, \bar{\mathbf{y}} \in \bar{S}_W(\mathbf{p})$ and $\mathbf{x}^* = \mathbf{W}^\dagger \mathbf{p}$. Put $R := \sqrt{\frac{\|\mathbf{p}\|_1}{k} - \|\mathbf{x}^*\|_2^2}$. Then $\|\bar{\mathbf{x}} - \mathbf{x}^*\|_2 = \|\bar{\mathbf{y}} - \mathbf{x}^*\|_2 = R$, and $\|\bar{\mathbf{y}} - \bar{\mathbf{x}}\|_2 \leq 2R$.*

Proof. From the definition of \mathbf{x}^* we have $(\bar{\mathbf{x}} - \mathbf{x}^*) \in \mathcal{N}(\mathbf{W})$, and $\mathbf{x}^* \perp (\bar{\mathbf{x}} - \mathbf{x}^*)$. Using Pythagoras' theorem and Lemma 1 yields

$$\|\bar{\mathbf{x}} - \mathbf{x}^*\|_2^2 = \frac{\|\mathbf{p}\|_1}{k} - \|\mathbf{x}^*\|_2^2 = R^2, \tag{5}$$

which means that any binary solution is on the hypersphere centered in \mathbf{x}^* with radius $\sqrt{\frac{\|\mathbf{p}\|_1}{k} - \|\mathbf{x}^*\|_2^2}$. Therefore,

$$\|\bar{\mathbf{x}} - \bar{\mathbf{y}}\|_2 \leq \|\bar{\mathbf{x}} - \mathbf{x}^*\|_2 + \|\bar{\mathbf{y}} - \mathbf{x}^*\|_2 = 2R. \tag{6} \quad \square$$

Corollary 1. *Let $\bar{\mathbf{x}}, \bar{\mathbf{y}} \in \bar{S}_W(\mathbf{p})$. Then $\|\bar{\mathbf{x}} - \bar{\mathbf{y}}\|_1 \leq 4(\frac{\|\mathbf{p}\|_1}{k} - \|\mathbf{x}^*\|_2^2)$.*

Proof. As $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ are binary, we have $\|\bar{\mathbf{x}} - \bar{\mathbf{y}}\|_1 = \|\bar{\mathbf{x}} - \bar{\mathbf{y}}\|_2^2$. The corollary now follows directly from Theorem 1. □

The norm $\|\bar{\mathbf{x}} - \bar{\mathbf{y}}\|_1$ corresponds to the number of pixels that are different between $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$. As one is typically mainly interested in the *fraction* of pixels that can be different, we introduce the *variability* of (\mathbf{W}, \mathbf{p}) , defined by $V := \frac{4R^2}{n}$, which is an upper bound for the fraction of pixels that are different between any two binary solutions.

4 A Bound on the Difference with a Particular Binary Image

The fact that all elements of $\bar{S}_W(\mathbf{p})$ have equal distance to the central reconstruction \mathbf{x}^* , combined with the facts that binary solutions are the shortest solutions among all integer solutions (Lemma 2) and that \mathbf{x}^* is the shortest real-valued solution, suggests that binary solutions can often be found near \mathbf{x}^* . It is therefore natural to consider the image that is obtained by rounding each entry of \mathbf{x}^* to the nearest binary value.

Let $\langle \alpha \rangle = \min(|\alpha|, |\alpha - 1|)$ for $\alpha \in \mathbb{R}$, and put $T = \sqrt{\sum_{i=1}^n \langle x_i^* \rangle^2}$, i.e., the Euclidean distance from \mathbf{x}^* to the nearest binary vector. Applying Theorem 1 yields

Corollary 2. *If $R < T$, then $\bar{S}_W(\mathbf{p}) = \emptyset$.*

If $R = T$, then all solutions in $\bar{S}_W(\mathbf{p})$ can be obtained by rounding the values in \mathbf{x}^ to the nearest binary values, and variations are only possible for the entries i where $x_i^* = \frac{1}{2}$.*

Let $\bar{\mathbf{r}} \in \{0, 1\}^n$ such that $\|\bar{\mathbf{r}} - \mathbf{x}^*\|_2 = T$, i.e., $\bar{\mathbf{r}}$ is among the binary vectors that are nearest to \mathbf{x}^* in the Euclidean sense. If $R > T$ and $R - T$ is small, it is possible to say that a fraction of the rounded values are correct, i.e., to provide an upper bound on the number of pixel differences between any solution in $\bar{S}_W(\mathbf{p})$ and $\bar{\mathbf{r}}$.

In most cases we can not say which rounded values are correct. Suppose that $\bar{\mathbf{x}} \in \bar{S}_W(\mathbf{p})$ and that $\bar{r}_i = 1$ whereas $\bar{x}_i = 0$. Note that we have $x_i^* \geq \frac{1}{2}$. Put $\tilde{\mathbf{r}} := \bar{\mathbf{r}}$ and then set \tilde{r}_i to 0. We then have $\|\tilde{\mathbf{r}} - \mathbf{x}^*\|_2^2 = \|\bar{\mathbf{r}} - \mathbf{x}^*\|_2^2 - |x_i^* - 1|^2 + |x_i^*|^2 = \|\bar{\mathbf{r}} - \mathbf{x}^*\|_2^2 + 2x_i^* - 1$. Similarly, if $\bar{r}_i = 0$, then the squared Euclidean distance increases by $1 - 2x_i^*$ by setting pixel i to 1. Each time an entry i of $\bar{\mathbf{r}}$ is changed, the squared Euclidean distance to \mathbf{x}^* increases by $b_i := |2x_i^* - 1|$.

As the Euclidean distance from \mathbf{x}^* to $\bar{\mathbf{x}}$ is R , a bound can now be derived on the maximal number of pixels in $\bar{\mathbf{r}}$ that must be changed to move from $\bar{\mathbf{r}}$ to $\bar{\mathbf{x}}$.

Let us order the values b_i ($i = 1, \dots, n$) such that $b_i \leq b_{i+1}$ for $1 \leq i \leq n - 1$. Assuming that $\bar{S}_W(\mathbf{p}) \neq \emptyset$, we have $R \geq \|\bar{\mathbf{r}} - \mathbf{x}^*\|_2$ and the change of s entries of $\bar{\mathbf{r}}$ would increase the distance between $\bar{\mathbf{r}}$ and \mathbf{x}^* such that $R^2 \geq \|\bar{\mathbf{r}} - \mathbf{x}^*\|_2^2 + \sum_{j=1}^s b_j$.

Theorem 2. *Let $\bar{\mathbf{r}}$, R and b_i ($i = 1, \dots, n$) be as defined above. Let $\bar{\mathbf{x}} \in \bar{S}_W(\mathbf{p})$ and $\mathbf{x}^* = \mathbf{W}^\dagger \mathbf{p}$. Choose s such that*

$$\sum_{i=1}^s b_i \leq R^2 - \|\bar{\mathbf{r}} - \mathbf{x}^*\|_2^2 < \sum_{j=1}^{s+1} b_j. \tag{6}$$

Then at most s pixels can have the wrong value in $\bar{\mathbf{r}}$ with respect to $\bar{\mathbf{x}}$ and at least $n - s$ pixels must have the correct value.

Proof. Due to the increasing order of the b_i 's, changing more than s pixels in $\bar{\mathbf{r}}$ will result in a vector $\tilde{\mathbf{r}}$ for which $\|\tilde{\mathbf{r}} - \mathbf{x}^*\|_2 > R$, which cannot be an element of $\bar{S}_W(\mathbf{p})$. \square

Corollary 3. *Let s be as in Theorem 2. Let $\bar{\mathbf{x}}, \bar{\mathbf{y}} \in \bar{S}_W(\mathbf{p})$. Then $\|\bar{\mathbf{x}} - \bar{\mathbf{y}}\|_1 \leq 2s$.*

In fact, Corollary 3 can be sharpened as follows. Theorem 2 bounds the number of pixel differences between $\bar{\mathbf{x}}$ and $\bar{\mathbf{r}}$, and between $\bar{\mathbf{y}}$ and $\bar{\mathbf{r}}$. When using these two bounds to determine an upper bound on the number of differences between $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$, we can assume that these two sets of pixel differences are disjoint, as otherwise the difference between $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ will only be smaller. This observation leads to the following corollary:

Corollary 4. *Let $\bar{\mathbf{r}}$ and b_i ($i = 1, \dots, n$) be as defined above. Let $\bar{\mathbf{x}}, \bar{\mathbf{y}} \in \bar{S}_W(\mathbf{p})$. Choose t such that*

$$\sum_{i=1}^t b_i \leq 2(R^2 - \|\bar{\mathbf{r}} - \mathbf{x}^*\|_2^2) < \sum_{j=1}^{t+1} b_j. \tag{7}$$

Then at most t pixels can be different between $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$.

5 Experiments and Results

A series of experiments was performed to investigate the practical value of the bounds given in Theorems 1 and 2 for a range of images. The experiments are all based on simulated projection data obtained by computing the projections of the test images (so-called *phantoms*) in Fig. 2: (1) a simple, nearly convex object; (2) a more complex object, containing a small hole; (3) a cross-section of a cylinder head from a combustion engine; (4) a cross-section of femur rat bone. All phantoms have a size of 512×512 pixels. For experiments with varying image size (smaller than 512×512), the phantoms have been downsampled to obtain binary images of the appropriate sizes.

In each experiment, the central reconstruction \mathbf{x}^* was first computed using the CGLS algorithm. Based on the central reconstruction, the variability V was computed, and $\bar{\mathbf{r}}$ was computed by rounding \mathbf{x}^* to the nearest binary vectors (choosing $\bar{r}_i = 1$ if $x_i^* = \frac{1}{2}$). The upper bound s from Theorem 2 on the number of differences between $\bar{\mathbf{r}}$ and the phantom image $\bar{\mathbf{x}}$ was then computed, followed by a bound on the fraction of pixel differences $U := \frac{s}{n}$, and the actual fraction of differences $E := \frac{e}{n}$, where e is the number of pixel differences between $\bar{\mathbf{r}}$ and $\bar{\mathbf{x}}$.

In the next subsections, the experimental results will be presented for the grid model and the strip model, respectively.

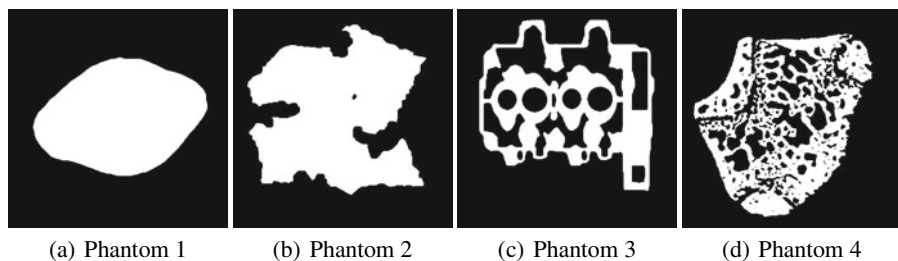


Fig. 2. Original phantom images used for the experiments

In the grid model, a projection direction is represented by a pair of integers $(a, b) \in \mathbb{Z}^2$, such that $\gcd(a, b) = 1$ and $a \geq 0$. Let \mathcal{A} be the set of all such pairs. For any positive integer M , put $\mathcal{A}_M := \{(a, b) \in \mathcal{A} : \max(a, |b|) = M\}$ and order the elements of \mathcal{A}_M , firstly by increasing value of a , secondly by increasing value of $|b|$, and thirdly by decreasing value of b . For example, $\mathcal{A}_3 = \{(1, 3), (1, -3), (2, 3), (2, -3), (3, 1), (3, -1), (3, 2), (3, -2)\}$. For any positive integer M , the ordered set \mathcal{D}_M is formed by concatenating $\mathcal{A}_1, \dots, \mathcal{A}_M$; for example, $\mathcal{D}_3 = \{(0, 1), (1, 0), (1, 1), (1, -1), (1, 2), (1, -2), (2, 1), (2, -1), (1, 3), (1, -3), (2, 3), (2, -3), (3, 1), (3, -1), (3, 2), (3, -2)\}$. To perform an experiment with k projection angles, the first k directions were selected from the set \mathcal{D}_{20} . This means that when the number of direction is increased, the old set of directions is always included in the new set of directions.

Experiments have been performed based on the three phantom images, scaled to sizes of 32×32 , 128×128 and 512×512 respectively, varying the number of projection

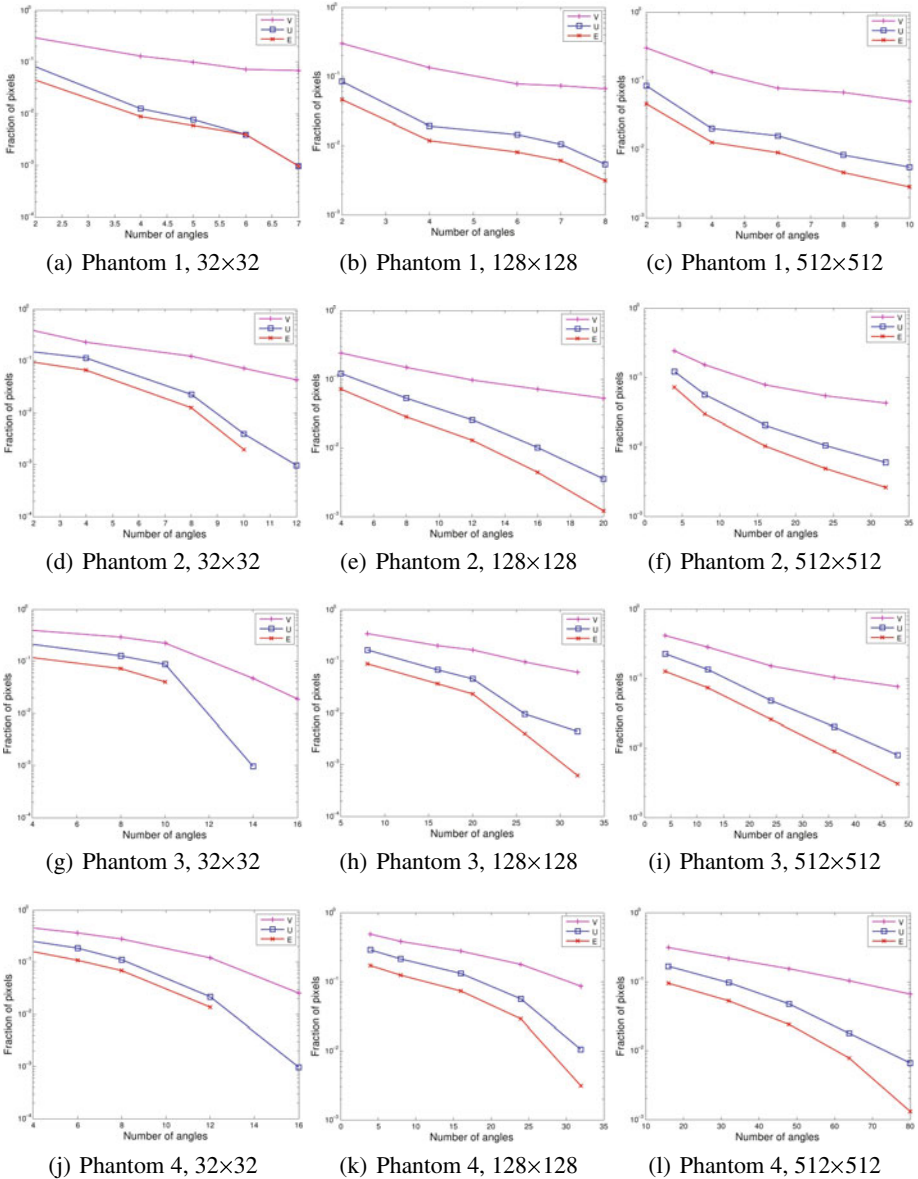


Fig. 3. Grid model: computed bounds as a function of the number of projection directions

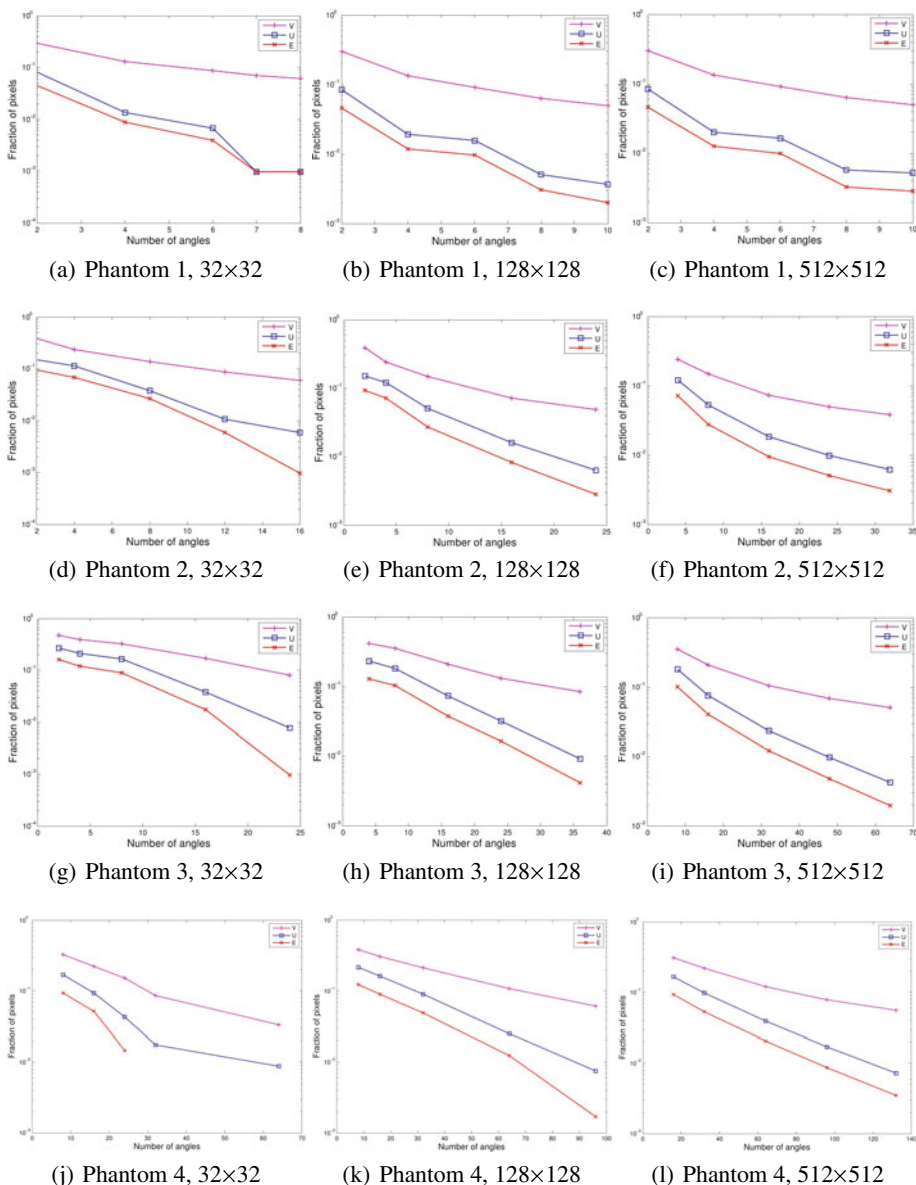


Fig. 4. Strip model: computed bounds as a function of the number of projection directions

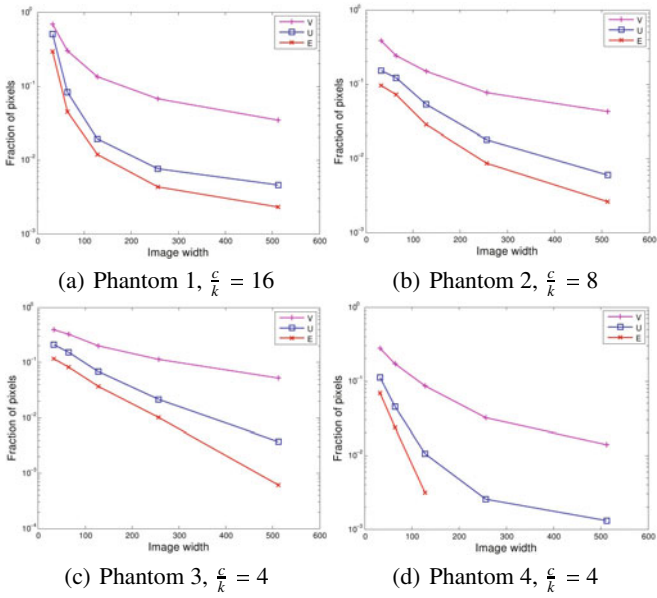


Fig. 5. Grid model: computed bounds as a function of the image width c , while keeping the ratio $\frac{c}{k}$ constant

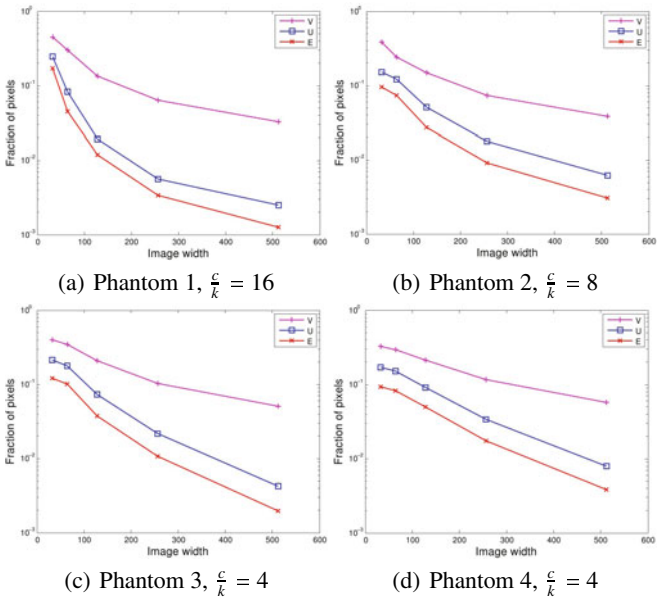


Fig. 6. Strip model: computed bounds as a function of the image width c , while keeping the ratio $\frac{c}{k}$ constant

directions. The results are shown in Fig. 3. In a second series of experiments, the size of the image was varied from 32×32 up to 512×512 . When increasing the image width c , the number of pixels grows quadratically with c , while the number of line sums per projection direction grows linearly with c . To compare results for different image sizes, the ratio $\frac{c}{k}$, between the image width and the number of projection directions is kept constant, increasing the number of projection directions along with the image size. The ratio $\frac{c}{k}$ is chosen separately for each phantom, such that it leads to values near $U = 0.01$ for sufficiently large images.

The experiments for the strip model have been performed using projection angles equally distributed between 0 and 180 degrees. Projections have been computed based on sets of parallel strips, each strip having a width that equals the pixel size.

Experiments have been performed based on the three phantom images, scaled to sizes of 32×32 , 128×128 and 512×512 respectively, varying the number of projection directions. The results are shown in Fig. 3. In a second series of experiments, the size of the image was varied from 32×32 up to 512×512 , while keeping the ratio $\frac{c}{k}$ constant, increasing the number of projection directions along with the image size. The ratio $\frac{c}{k}$ is chosen separately for each phantom, such that it leads to values near $U = 0.01$ for sufficiently large images.

Despite the fact that the four phantoms have strong differences in shape and morphology, and that the grid and strip models are quite different, the results shown in Figs. 3-6 are surprisingly consistent throughout all experiments. Clearly, both the variability V and the upper bound U on the fraction of pixel differences between \bar{r} and x^* become smaller as the number of projection directions is increased. We see that the phantom \bar{x} is typically much closer to \bar{r} than to x^* , and that the actual fraction of differences between these two images is approximated quite sharply by the upper bound U . In Fig. 3(d), (g) and (j), and Fig. 4(j), part of the graph E is missing. In fact, in these cases E is zero, such that they cannot be displayed in the logarithmic scale.

In the strip model, the total number of measured line sums is given by ck , whereas the total number of unknown pixels is c^2 . Fig. 6 illustrates that even if $\frac{c}{k} \geq 4$, the bound from Theorem 2 leads to a guarantee that the fraction of pixel differences between \bar{x} and \bar{r} is of the order of 1%, so \bar{r} is a very good approximation of the original binary image.

6 Outlook and Conclusions

In this article, we have presented two general bounds on the accuracy of reconstructions in binary tomography, with respect to the unknown original object. The bounds can be computed efficiently and give guarantees on the number of pixels that can be different between any two binary reconstructions that satisfy given line sums, and on the difference between a particular binary image, obtained by rounding the central projection to the nearest binary vector, and any binary image satisfying the projections. The experimental results show that by using these bounds, one can prove that the number of differences between binary reconstructions must be very small, even when the corresponding real-valued system of equations is severely underdetermined. In order to make these bound practically useful, our results will have to be extended to deal with noisy projection data, which we will incorporate in future research.

Acknowledgement. L.H. was supported by the OTKA grants K67580 and K75566, and by the TÁMOP 4.2.1./B-09/1/KONV-2010-0007 project. The project is implemented through the New Hungary Development Plan, cofinanced by the European Social Fund and the European Regional Development Fund. W.F. acknowledges support from the Erasmus Mundus program of the European Union.

References

1. Alpers, A.: Instability and Stability in Discrete Tomography. Ph.D. thesis, Technische Universität München. Shaker Verlag, Aachen (2003) ISBN 3-8322-2355-X
2. Alpers, A., Brunetti, S.: Stability results for the reconstruction of binary pictures from two projections. *Image and Vision Computing* 25(10), 1599–1608 (2007)
3. Alpers, A., Gritzmann, P.: On stability, error correction, and noise compensation in discrete tomography. *SIAM Journal on Discrete Mathematics* 20(1), 227–239 (2006)
4. Ben-Israel, A., Greville, T.N.E.: Generalized inverses: Theory and applications. Canadian Math. Soc. (2002)
5. Hajdu, L., Tijdeman, R.: Algebraic aspects of discrete tomography. *J. Reine Angew. Math.* 534, 119–128 (2001)
6. Herman, G.T.: Fundamentals of Computerized Tomography: Image reconstruction from projections. Springer, Heidelberg (2009)
7. Herman, G.T., Kuba, A. (eds.): Discrete Tomography: Foundations, Algorithms and Applications. Birkhäuser, Boston (1999)
8. Herman, G.T., Kuba, A. (eds.): Advances in Discrete Tomography and its Applications. Birkhäuser, Boston (2007)
9. Jinschek, J.R., Batenburg, K.J., Calderon, H.A., Kilaas, R., Radmilovic, V., Kisielowski, C.: 3-D reconstruction of the atomic positions in a simulated gold nanocrystal based on discrete tomography. *Ultramicroscopy* 108(6), 589–604 (2007)
10. Midgley, P.A., Dunin-Borkowski, R.E.: Electron tomography and holography in materials science. *Nature Materials* 8(4), 271–280 (2009)
11. Saad, Y.: Iterative Methods for Sparse Linear Systems. SIAM, Philadelphia (2003)
12. Van der Sluis, A., Van der Vorst, H.A.: SIRT and CG-type methods for the iterative solution of sparse linear least-squares problems. *Linear Algebra Appl.* 130, 257–302 (1990)
13. Van Dalen, B.: On the difference between solutions of discrete tomography problems. *Journal of Combinatorics and Number Theory* 1, 15–29 (2009)
14. Van Dalen, B.: On the difference between solutions of discrete tomography problems II. *Pure Mathematics and Applications* 20, 103–112 (2009)
15. Van Dalen, B.: Stability results for uniquely determined sets from two directions in discrete tomography. *Discrete Mathematics* 309, 3905–3916 (2009)
16. Zhua, J., Li, X., Ye, Y., Wang, G.: Analysis on the strip-based projection model for discrete tomography. *Discrete Appl. Math.* 156(12), 2359–2367 (2008)

Tiling the Plane with Permutations

Alexandre Blondin Massé¹, Andrea Frosini²,
Simone Rinaldi³, and Laurent Vuillon¹

¹ Laboratoire de Mathématiques, Université de Savoie, CNRS UMR 5127,
73376 Le Bourget du Lac

² Università di Firenze, Dipartimento di Sistemi e Informatica,
viale Morgagni 65, 50134 Firenze

³ Università di Siena, Dipartimento di Matematica e Informatica,
Pian dei Mantellini 44, 53100 Siena

Abstract. A *permutomino* is a polyomino uniquely determined by a pair of permutations. Recently permutominoes, and in particular *convex permutominoes* have been studied by several authors concerning their analytical and bijective enumeration, tomographical reconstruction, and the algebraic characterization of the associated permutations [23]. On the other side, Beauquier and Nivat [5] introduced and gave a characterization of the class of *pseudo-square* polyominoes, i.e. polyominoes that tile the plane by translation: a polyomino is called pseudo-square if its boundary word may be factorized as $XY\overline{X}\overline{Y}$.

In this paper we consider the pseudo-square polyominoes which are also convex permutominoes. By using the Beauquier-Nivat characterization we provide some geometrical and combinatorial properties of such objects, and we show for any fixed X , each word Y such that $XY\overline{X}\overline{Y}$ is pseudo-square is prefix of an infinite word Y_∞ with period $4|X|_N |X|_E$.

Some conjectures obtained through exhaustive search are also presented and discussed in the final section.

1 Introduction

Tiling the plane using polyominoes is a rather popular research topic, sometimes related with computational theory, mathematical logic and discrete geometry. This subject has been studied under different points of view: combinatorial properties of the tiles have been considered in order to count them or the patterns they produce in the plane, and, consequently, group them in classes presenting similar behaviors; a geometrical approach has been applied in order to find new characterizations of the tiles in terms of their shape or properties of their border; finally, also tomographical aspects of tilings have been studied with the aim of retrieving geometrical properties of a tiling from partial and sometimes inaccurate measurements of the local densities (see [6]).

Interesting results have been achieved by restricting the class of sets of tiles only to those having one single element. In particular Wijshoff and Van Leeuwen [8] considered the *exact polyominoes* (i.e. polyominoes which tile the plane by translation) and proved that the problem of recognizing them is decidable. In [5],

Beauquier and Nivat studied the same problem from a purely geometrical point of view and they found a characterization of all the exact polyominoes by using properties of the words which describe their boundaries. In particular they stated that the boundary word coding these polyominoes shows a pattern $XYZ\overline{X}\overline{Y}\overline{Z}$, called a *pseudo-hexagon*, where one of the variable may be empty in which case the pattern $XY\overline{X}\overline{Y}$ is called a *pseudo-square*. Successively, in [1] the authors studied some combinatorial and enumeration problems on the pseudo-square convex polyominoes.

In this paper we relate pseudo-square polyominoes with the class of permutominoes. A *permutomino* of size n is a polyomino determined by particular pairs (π_1, π_2) of permutations of length n , such that $\pi_1(i) \neq \pi_2(i)$, for $1 \leq i \leq n$ (see, for instance, Fig. 4). An equivalent definition for a permutomino is that, for each abscissa (ordinate) between 1 and n , there is exactly one vertical (horizontal) side in the boundary with that coordinate.

Permutominoes can be viewed as special types of *permutation diagrams*, and they have been introduced by Kassel et al. [7] while studying some algebraic problems related with the \tilde{R} -polynomials associated with a pair (π_1, π_2) of permutations.

During the last years, a particular class of permutominoes, namely the class of convex permutominoes have been widely studied: in [3] the authors provide their enumeration according to the size, while in [2] the authors give a characterization of permutation defining convex permutominoes.

In this paper we consider the class of *pseudo-square convex permutominoes* (briefly, *psc-permutominoes*), i.e. polyominoes which are both pseudo-square and convex permutominoes. This problem raises interesting properties since it relates to combinatorics on words and classical problems on polyominoes. Moreover these objects are uniquely defined by permutations, thus they give us an effective way to tile the plane using permutations. Hence this problem shows strict relations with the vast combinatorics on pattern avoiding permutations [2].

In this paper we focus on the problem of establishing if, for a given word X , there is at least a word Y which is *compatible* with X , that is, $XY\overline{X}\overline{Y}$ represents a *psc-permutomino*. Using the Beauquier-Nivat characterization, and the results from [1], we show that for any given X there are at most two different words of minimal length Y and Y' compatible with X . These words are such that $|Y| = |Y'|$, and Y' can be obtained from Y by means of renaming of words. Moreover, they individuate the *growing direction* of the permutomino, so we distinguish among the *up* and the *down-growing* direction of a *psc-permutomino* and study these two cases separately.

In studying the *psc-permutominoes* with up-growing direction, we prove that X uniquely determines an infinite word Y_∞ , and that every word Y which is up-compatible with X (i.e. $XY\overline{X}\overline{Y}$ is a *psc-permutomino* with up-growing direction) is a prefix of Y_∞ . Moreover we show that Y_∞ has period $4|X|_E|X|_N$. The same results can be reformulated in the case of down-growing direction.

Several challenging problems remain open, and some conjectures obtained through exhaustive search are also presented and discussed in the final section.

2 Two Classes of Polyominoes

In the plane $\mathbb{Z} \times \mathbb{Z}$ a *cell* is a unit square whose vertices have integer coordinates, and a *polyomino* is a finite connected union of cells having no cut point (see Fig. 1(a)). Polyominoes are defined up to translations, and we deal with polyominoes without “holes”, i.e. polyominoes whose boundary is a single loop. A *column* (resp. *row*) of a polyomino is the intersection between the polyomino and an infinite strip of cells whose centers lie on a vertical (resp. horizontal) line. A polyomino is said to be *column-convex* (resp. *row-convex*) when its intersection with any vertical (resp. horizontal) line is connected. A polyomino is convex if it is both column and row convex (see Fig. 1(b,c)).

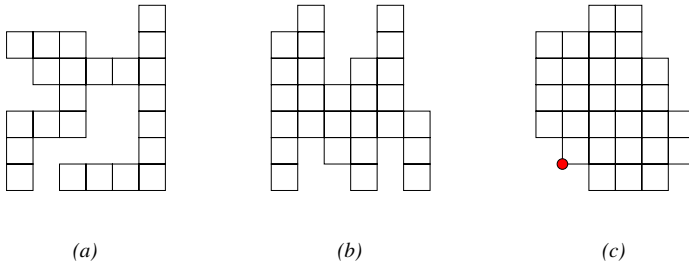


Fig. 1. A polyomino (a), a column-convex polyomino (b), a convex polyomino (c)

A particular subclass of the class of convex polyominoes consists of the *parallelogram* polyominoes, defined by two lattice paths that use north (vertical) and east (horizontal) unitary steps, and intersect only at their origin and extremity. These paths are commonly called the *upper* and the *lower path* (see Fig. 2).

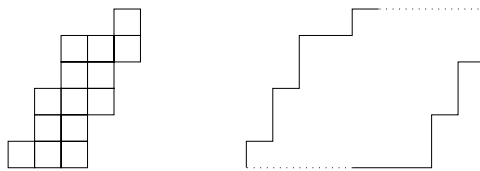


Fig. 2. A parallelogram polyomino, its upper and lower paths

The boundary of a polyomino can be conveniently represented by a boundary word defined on the alphabet $\Sigma = \{N, S, E, W\}$, where *N* (resp. *E*, *S*, *W*) stands for the north (resp. east, south, west) unit step. The word representing a polyomino is obtained simply by following its boundary from a starting point in a clockwise orientation. For instance, the polyomino in Fig. 1(c), starting from the highlighted point, is represented by the word

$$X = \text{NWNNNNEENEESSESSSESSSWSWWWNW.}$$

Moreover, if $X = x_1x_2 \dots x_r$ is a word, where $x_i \in \{N, E, S, W\}$, then the complement $\overline{X} = \overline{x}_r \dots \overline{x}_2\overline{x}_1$ is defined by $\overline{N} = S$, $\overline{S} = N$, $\overline{W} = E$, and $\overline{E} = W$. We define the length of a word X by $|X| = r$, and by $|X|_{x_i}$ the number of occurrences of the letter x_i in X . We observe that in any polyomino the length of the boundary word coincides with its perimeter.

2.1 Polyominoes That Tile the Plane

In [5], Beauquier and Nivat studied the class of exact polyominoes, i.e. polyominoes that tile the plane by translation. They found a characterization of all the exact polyominoes by using properties of the words which describe their boundaries. In particular they stated that the boundary words coding these polyominoes show a pattern $XYZ\overline{X}\overline{Y}\overline{Z}$, called a *pseudo hexagon*, where one of the variable may be empty in which case the pattern $XY\overline{X}\overline{Y}$ is called a *pseudo-square*.

In the rest of the paper, we will deal with pseudo-square convex polyominoes (briefly *psc-polyominoes*) As an example, the polyomino in Fig. 3 (a) is a *psc-polyomino*, and the decomposition of its boundary word is

$$X = NNENEEN, Y = EE\overline{N}E, \overline{X} = \overline{N} \overline{E} \overline{E} \overline{N} \overline{E} \overline{N} \overline{N} \text{ and } \overline{Y} = \overline{E} \overline{N} \overline{E} \overline{E}.$$

We recall some properties about *psc-polyominoes* obtained in [1], which will be useful through the paper. We first consider pseudo-square parallelogram polyominoes. In this case we have:

Proposition 1. *If $XY\overline{X}\overline{Y}$ is a decomposition of the boundary word of a pseudo-square parallelogram polyomino, then XY encodes its upper path, and YX its lower path. Moreover, X starts and ends with N , and Y starts and ends with E .*

For *psc-polyominoes* which are not parallelograms, we can have at most two decompositions which are strictly related to each other:

1. the first one is such that the path X of the decomposition starts from the lowest point in the leftmost column and ends in the leftmost point in the uppermost row (see Figure 3 (c)), then it has the form $N^+(E \vee N)^*N^+$. The starting point of such decomposition is called A ; we observe that also parallelogram polyominoes have this kind of decomposition;
2. the second one is such that the path X starts from the uppermost point in the leftmost column and ends in the rightmost point in the uppermost row, (see Figures 3 (b), (d)). The starting point of such decomposition is called B .

We also observe that each *psc-polyomino* with a decomposition of the second type can be transformed – under an horizontal reflection – into another *psc-polyomino* whose decomposition is of the first type (note that the two polyominoes may coincide). So, without loss of generality we will only consider *psc-polyominoes* of with a decomposition of the first type, starting from A . Hence, from now on, the word X encoding the boundary of a generic *psc-permutomino* is assumed to have the form $N^+(E \vee N)^*N^+$.

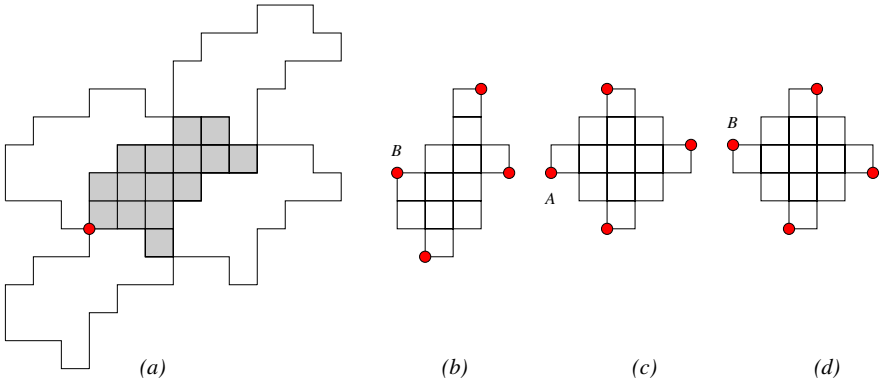


Fig. 3. (a) a *psc*-polyomino (with highlighted cells) having only a decomposition of the first type, and the corresponding tiling of the plane; (b) a *psc*-polyomino having one decomposition of the second type; in (c), (d) a *psc*-polyomino admitting decompositions of the first and of the second type; the starting points are *A* and *B*, respectively

2.2 Polyominoes Determined by Permutations

Let P be a polyomino without holes, having n rows and n columns, $n \geq 1$. Let $\mathcal{A} = (A_1, \dots, A_{2(r+1)})$ be the list of its vertices (i.e., corners of its boundary) ordered in a clockwise direction starting from the lowest leftmost vertex, with $A_i = (x_i, y_i)$. We say that P is a *permutomino* if $\mathcal{P}_1 = (A_1, A_3, \dots, A_{2r+1})$ and $\mathcal{P}_2 = (A_2, A_4, \dots, A_{2r+2})$ represent two permutations of length n . Obviously, if P is a permutomino, then $r = n$, see Fig. 4. As an immediate consequence we have that $m = n$, and so a permutomino has n rows and n columns (n will be called its *size*), and the number of its vertices is $2(n + 1)$. The two sets \mathcal{P}_1 and \mathcal{P}_2 can be regarded as two permutation matrices of $[n + 1] = \{1, 2, \dots, n + 1\}$ having no common points; we indicate the permutations associated with them by π_1 and π_2 , respectively (see Fig. 4). They are called the *permutations associated with P* .

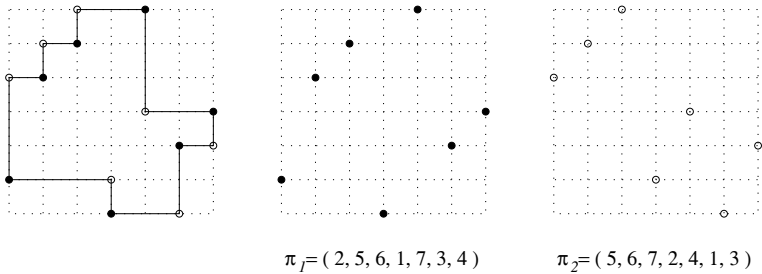


Fig. 4. A permutomino and the two associated permutations. The dotted black vertices represent the elements of \mathcal{P}_1 , while the circled vertices the elements of \mathcal{P}_2 .

From the definition any permutomino P of size n has the property that, for each abscissa (ordinate) between 1 and n there is exactly one vertical (horizontal) side in the boundary of P with that coordinate. It is simple to observe that this property is also a sufficient condition for a polyomino to be a permutomino.

3 Convex Permutominoes Tiling the Plane

In this section we study convex permutominoes which are also pseudo-square polyominoes, that we call *pseudo-square convex permutominoes* (briefly, *psc-permutominoes*). They present interesting geometrical and combinatorial aspects.

First, we recall that we assume that the word X encoding the boundary of a generic *psc*-permutomino has the form $N^+(E \vee N)^*N^+$. Moreover, we will work indifferently with the word or the path representation of X and Y .

We say that a binary word X is *compatible* with Y if the word $XY\overline{X}\overline{Y}$ represents the boundary of a *psc*-polyomino. So, for instance, we have that the word $X = NNENN$ is compatible with $Y = EENEE$ (see Fig. 5 (a)), with $Y' = EESEE$ (see Fig. 5 (b)), and with $Y'' = EESEESSEESSEESSEESSE$ (see Fig. 5 (c)). A word X of the form $N^+(E \vee N)^*N^+$ is said to be *exact* if there is at least a word Y which is compatible with X .

From the definition of permutomino and of pseudo-square, and from the convexity constraint, The following statement holds.

Proposition 2. *For each word X of the form $N^+(E \vee N)^*N^+$ there exist at most two different words Y and Y' , having minimal length, compatible with X such that:*

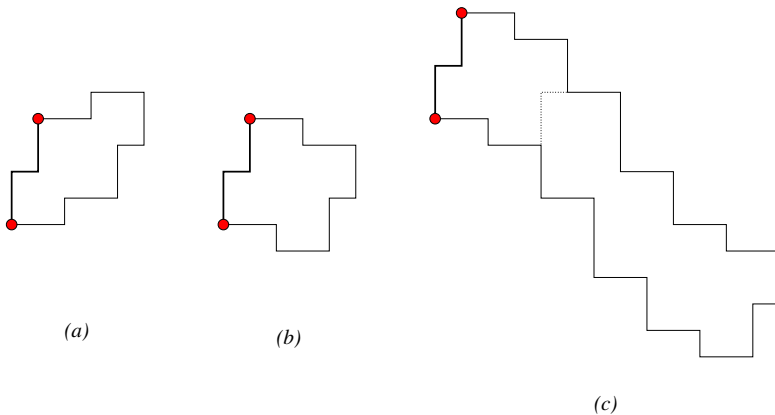


Fig. 5. Three *psc*-permutominoes obtained from the same word $X = NNENN$. The permutomino in (a) has an upward growing direction, while those in (b) and (c) have a downward the growing direction.

1. neither Y nor Y' is a prefix of the other,
2. if there exists Y'' compatible with X , then either Y or Y' is a prefix of Y'' .

It is also clear that, in a pseudo-square convex permutomino, the word X completely determines the related words Y and Y' . For instance referring to Figure 5, the word $X = NNENN$ is compatible with $Y = EENEE$, and $Y' = EESEE$ while $Y'' = Y'SSEESSSEESSY'$.

The first classification of *psc*-permutominoes is given by the concept of *growing direction*, which follows directly from Proposition 2. We say that a *psc*-permutomino has *upward* growing direction if the word Y has the form $E^+(N \vee E)^*E^+$. Similarly, we say that it has *downward* growing direction if Y has the form $E^+(S \vee E)^*E^+$ (see Fig. 5). In the next sections we will study separately the *psc*-permutominoes having upward and downward growing direction. According to this, we say that X and Y are *up-compatible* (resp. *down-compatible*) with X if $XY\overline{X}Y$ is a *psc*-permutomino having upward (resp. downward) growing direction.

3.1 Up-Growing Direction: Pseudo-square Parallelogram Permutominoes

The reader may have noticed that *psc*-permutominoes having upward growing direction are precisely the pseudo-square parallelogram permutominoes (see Figure 5(a)). We observe that, starting from X , and given a positive integer n , there is a unique path $Y(n)$ of length n , and using steps E and N , such that the object $X(n)$ obtained by concatenating $Y(n)$ at the beginning and at the end of X has exactly one side for each abscissa and ordinate between 1 and n . Similarly, letting n tend to infinity, we obtain a unique infinite path Y_∞ , and an object $X(\infty)$ having exactly one side for each abscissa and ordinate greater than 1 (see Fig. 6). In $X(n)$, the path Y_∞ starting from the end (resp. beginning) of X will be called the *upper* (resp. *lower*) path of $X(n)$. The following property is straightforward.

Proposition 3. *Every word Y which is up-compatible with X is a prefix of Y_∞ .*

In what follows, we prove that the word Y_∞ is periodic, and we provide a simple algorithm to determine a prefix of length n of Y_∞ from X .

First, we start by recalling the definition of *mex* of positive (or empty) integer set v , noted $\text{mex}(v)$: it is the smallest positive integer of the complementary of the set v in \mathbb{N} . For instance, if $v = \{0, 1, 4\}$ then the complement of v is $\{2, 3, 5, 6, \dots\}$ thus $\text{mex}(v) = 2$. This definition appears in some works of A.S. Fraenkel and it is related to the Nim game, Sturmian and balanced sequences 4.

The basic idea for constructing the path Y_∞ is to look separately at the horizontal and vertical steps in X .

The first step consists in building the vector V whose i th entry V_i is given by the length of the i th vertical segment in X , and the vector H , whose i th entry is given by the length of the i th horizontal segment in Y . So, for instance, if $X = NNENN$ we have $V = (2, 2)$ and $H = (1)$, see Fig. 6(a).

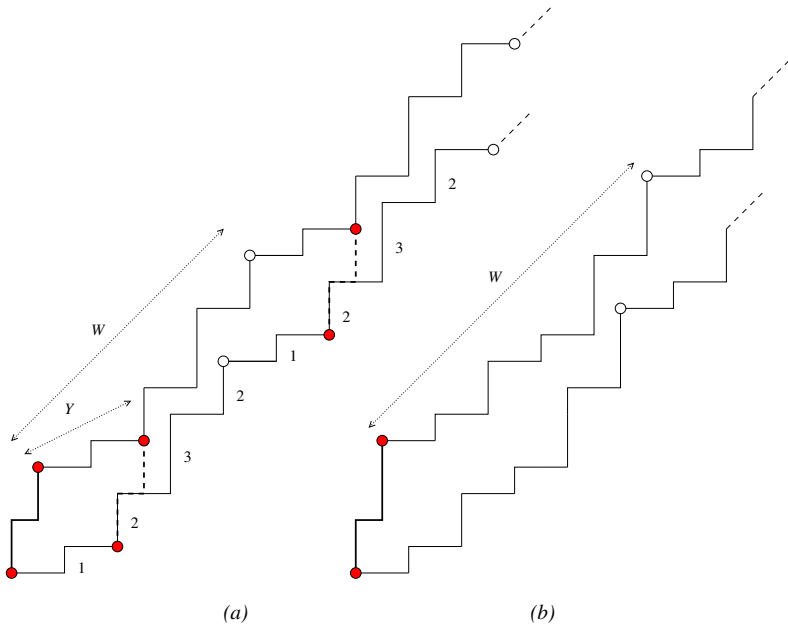


Fig. 6. The path $X(\infty)$ does not cross itself: (a) we have no pseudo-square parallelogram permutomino; (b) there is a word Y of length less than the period W of Y_∞ such that X is up-compatible with Y , and we have a pseudo-square parallelogram permutomino

Then we focus on the vertical steps growth, and for any given $i \geq 1$ we construct the integer set $v(i)$ by iteration, according to the following rule:

Base: $v(1) = \{0, V_1, V_1 + V_2, V_1 + V_2 + V_3, \dots\}$: it is the set of the ordinates where the vertical segments of X start or end; moreover, we set the variable $p(1) = 0$.

Step: $v(i + 1) = v(i) \cup \text{mex}(v(i)) \cup \text{mex}(v(i)) - p(i) + \max(v(i))$, then we set $p(i + 1) = \text{mex}(v(i))$.

We also build the integer vector V_Y , whose i th entry is given by the term $\text{mex}(v(i - 1)) - p(i - 1)$. Referring to our example, we have:

- $v(1) = \{0, 2, 4\}$, and $p(1) = 0$;
- $\text{mex}(v(1)) = 1$, $\text{mex}(v(1)) - p(1) + \max(v(1)) = 5$, that is $v(2) = \{0, 1, 2, 4, 5\}$, and $V_Y = (1)$; now, $p(2) = \text{mex}(v(1)) = 1$;
- $\text{mex}(v(2)) = 3$ and $\text{mex}(v(2)) - p(2) + \max(v(2)) = 3 - 1 + 5 = 7$, then $v(3) = \{0, 1, 2, 3, 4, 5, 7\}$, and $V_Y = (1, 2)$; now $p(3) = 3$;
- $\text{mex}(v(3)) = 6$ and $\text{mex}(v(3)) - p(3) + \max(v(3)) = 6 - 3 + 7 = 10$, that is $v(4) = \{0, 1, 2, 3, 4, 5, 6, 7, 10\}$, and $V_Y = (1, 2, 3)$; now, $p(4) = 6$;
- $\text{mex}(v(4)) = 8$ and $\text{mex}(v(4)) - p(4) + \max(v(4)) = 12$, that is $v(5) = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 12\}$ and $V_Y = (1, 2, 3, 2)$.

We observe that the i -th element of V_Y is just the length of the i th vertical segment added in the construction of Y_∞ (see Fig. 6 (a)). Now if we go on with the iteration, we observe that, for any $i > 4$, $V_Y(i) = V_Y(j)$, where j is simply the rest of the division of i by 4. This implies that V_Y is periodic of period 4. This means that the vertical segments in Y_∞ are such that the length of the i -th vertical segment is equal to $V_Y(i \bmod 4)$.

Starting from H , and with an analogous algorithm, we can recursively build the set $h(i)$, and then the vector H_Y , whose i th entry gives the length of the i th horizontal segment in Y_∞ . Referring to our example, we have that H_Y is periodic of length 2, precisely $H_Y = (2)$. This means that all the horizontal steps in Y_∞ have length 2 (see Fig. 6 (a)).

Using the previously defined tools, we are now ready to prove our main result.

Theorem 1. *For any given $X = N^+(E \vee N)^*N^+$, there exists a positive integer p dividing $4|X|_N|X|_E$ such that the infinite word Y_∞ has period p .*

Proof. To prove the periodicity of Y_∞ it is sufficient to prove that the vectors V_Y and H_Y are periodic. First of all, let us make some remarks on the construction of $v(i)$ previously described. We recall that $p(i) = \text{mex}(v(i - 1))$ with the convention that $\text{mex}(v(0)) = 0$. Thus there is an invariant in the construction: $\text{mex}(v(i)) - p(i) + \max(v(i)) - \text{mex}(v(i)) = \max(v(i)) - \text{mex}(v(i - 1))$. In fact, by construction we have $\max(v(i)) = \text{mex}(v(i - 1)) - p(i - 1) + \max(v(i - 1))$, then

$$\begin{aligned} \max(v(i)) - \text{mex}(v(i - 1)) &= -p(i - 1) + \max(v(i - 1)) \\ &= \max(v(i - 1)) - \text{mex}(v(i - 2)) \\ &= \max(v(1)) - \text{mex}(v(0)) = \max(v(1)). \end{aligned}$$

In other words, at each step $i > 0$ of the iteration we add to $v(i)$ two elements at distance $\max(v(1))$. In the path Y_∞ that we are building, these two elements represent a vertical segment in the lower path, and a vertical segment in the upper path, respectively.

Now we would like to encode the set $v(i)$ by means of a vector $w(i)[0, \dots, \max(v(i))]$ whose entries are the symbols $\{a, b, c, \cdot\}$, using the following:

- i. for every element j in the base set $v(1)$ we place the symbol a in position j of $w(1)$, and all other places are filled with \cdot ,
- ii. at the generic step $i + 1$ of the iteration, the newly added element $\text{mex}(v(i))$ is represented by placing b in position $\text{mex}(v(i))$ of $w(i + 1)$, while the element $\text{mex}(v(i)) - p(i) + \max(v(i))$ is represented placing c in position $\text{mex}(v(i)) - p(i) + \max(v(i))$ of $w(i + 1)$. We use the symbol \cdot to say that a given position has not been filled yet by a b .

Thus, at each step we add two new symbols, b and c , to the word $w(i - 1)$, and they are always at distance $\max(v(1))$.

So referring to our example, $v(1)$ is represented by the word $w(1) = a \cdot a \cdot a$, then the set $v(2) = \{0, 1, 2, 4, 5\}$ is represented by $aba \cdot ac$, since we add a

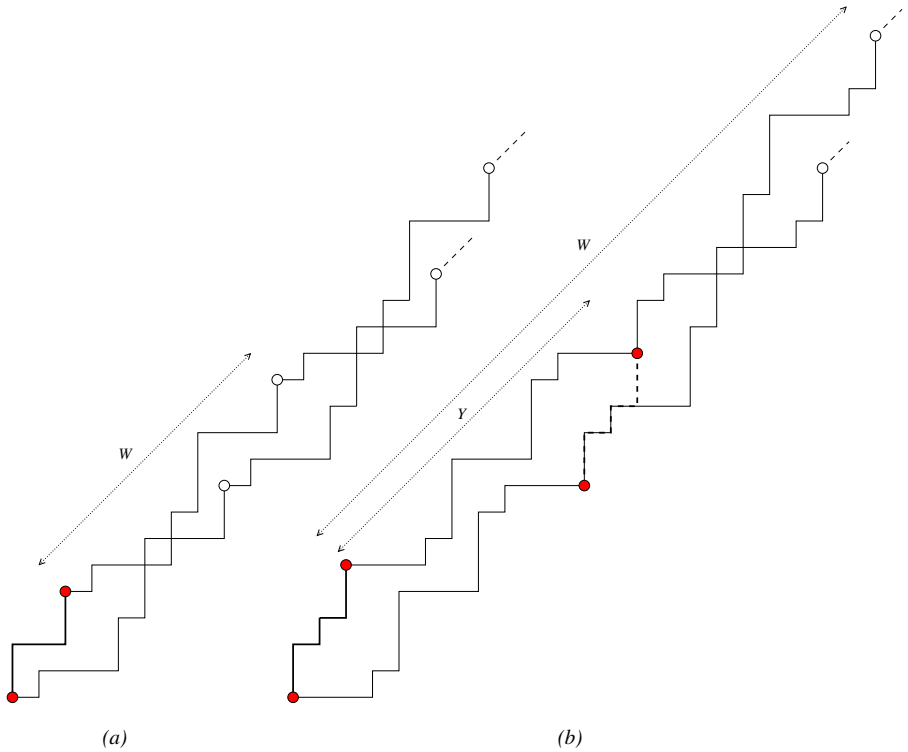


Fig. 7. The path $X(\infty)$ crosses itself: (a) there is no pseudo-square parallelogram permutomino; (b) there is a word Y of length less than the period W of Y_∞ such that X is up-compatible with Y , and we have a pseudo-square parallelogram permutomino

b in position 1 and a c in position 5, at distance $\max(v(1)) = 4$. Then the representation for $v(3) = \{0, 1, 2, 3, 4, 5, 7\}$ is $w(3) = ababac \cdot c$. Successively we have $w(4) = abababc \cdot c$, $w(5) = ababacbc \cdot c \cdot c$, and so on.

We can prove now that the element $2 \max(v(1))$ always represents a segment in the lower path and it is coded by b . Otherwise, by the invariant $2 \max(v(1)) - \max(v(1)) = \max(v(1))$ should represent a vertical segment of the lower path, and this is impossible because $\max(v(1))$ is clearly coded by a .

Using the same argument: each a of the word $w(1)$ in position $j > 0$ is “translated” by $\max(v(1))$ to a b in position $j + \max(v(1))$; each b is position $j \geq 0$ of $w(i)$ is translated to a c in position $j + \max(v(1))$, and similarly each c is position $j \geq 0$ of $w(i)$ is translated to a b in position $j + \max(v(1))$.

Thus, it’s when the element in position $2 \max(v(1))$ is filled by a b for the first time that we find the period. Indeed, let i be the first index such that $w(i)$ has a b in position $2 \max(v(1))$. Concerning the word $w(i)$ we have a in position 0, a in position $\max(v(1))$ and b in position $2 \max(v(1))$. For all other positions, a in position j implies b in position $j + \max(v(1))$, and c in position $j + 2 \max(v(1))$.

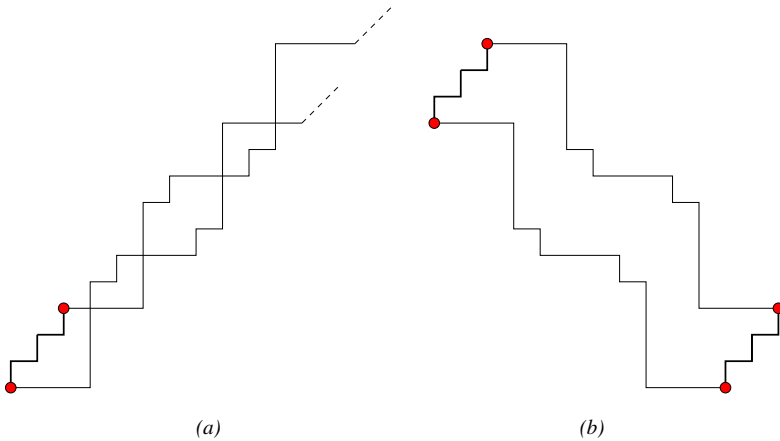


Fig. 8. (a) the path Y_∞ : there is no word Y which is up-compatible with X ; (b) a prefix Y' of the path Y'_∞ which is down-compatible with X

By construction the b in position $2 \max(v(1))$ is the last b placed in $w(i)$, and all the positions on the right of it are either filled by a c or by a dot. Moreover, all the other c in position greater than $2 \max(v(1))$ have been translated from an a in position j by the translation $2 \max(v(1))$. Finally, there is a clear correspondence between the elements of $w(1)$ and those of $w(i)[2 \max(v(1)), \dots, \max(v(i))]$, which leads to the periodicity. Referring to our example, we have $w(1) = a \cdot a \cdot a$, then $w(5) = abababc|b \cdot c \cdot c$. In summary, we need $2|X|_N$ iterations to have periodicity on V_Y of the vertical steps, while similarly, we need $2|X|_E$ iterations to have periodicity on H_Y . By mixing the two words, we have that Y_∞ has period $2|X|_N \cdot 2|X|_E = 4|X|_N|X|_E$. \square

The proof of Theorem 11 gives us a simple recursive way to generate the infinite word Y_∞ :

- $Y_\infty(0) = \emptyset$;
- $Y_\infty(i) = Y_\infty(i - 1) E^{H_Y(i) \bmod |H_Y|} N^{V_Y(i) \bmod |V_Y|}$.

Then Y_∞ is the limit of $Y_\infty(i)$ for $i \rightarrow \infty$.

From now on, let X be given, let p be the period of Y_∞ , and let $W = Y_\infty[1, \dots, p]$. We remark that the previously described algorithm gives us as a corollary a simple way to check if the boundary of $X(\infty)$ crosses itself or not. Since Y_∞ is periodic, it is sufficient to check it for at most p steps. Summarizing, concerning pseudo-square parallelogram permutominoes, we may have the following possibilities:

1. the boundary of $X(\infty)$ crosses itself. Then by Theorem 11 it crosses itself infinitely many times. In this case, we may have the two possibilities: (a) we have no pseudo-square parallelogram permutomino (see Fig. 7 (a)); (b) there is a prefix Y of W which is up-compatible with X , and we have a pseudo-square parallelogram permutomino (see Fig. 7 (b)).

2. the boundary of $X(\infty)$ does not cross itself; in this case, it may happen that there are no words Y which are up-compatible with X (see Fig. 6 (a)); otherwise, if there is at least a word Y up-compatible with X , then there are infinitely many such words; assuming that Y is the word having minimal length among these ones, then they have the form W^*Y (see Fig. 6 (b), where the black points identify the paths Y, WY, W^2Y, \dots , and the white points identify the paths W, W^2, \dots).

4 Down-Growing Direction

Most of the considerations made for the up-growing direction can be easily reformulated for the down-growing direction. Also in this case, for a given word X of the form $N^+(E \vee N)^*N^+$, we can define an infinite word Y'_∞ , and every word Y' which is down-compatible with X is a prefix of Y'_∞ . Moreover, we can write down an algorithm – completely analogous to that presented in the previous section – to recursively build the word Y'_∞ . There is also a strict relation between the two words Y_∞ and Y'_∞ which is exploited in the next statement.

Proposition 4. *Let X be a word of the form $N^+(E \vee N)^*N^+$. The following properties hold:*

- i) Y'_∞ is obtained from Y_∞ by changing all N steps with S steps (see Fig. 8).
- ii) The word Y'_∞ is periodic, and its period is a divisor of $4|X|_N|X|_E$.
- iii) If X is up-compatible with Y , then X is down-compatible with the word Y' , which is obtained from Y by changing all N steps with S steps (see Fig. 5).

The converse of iii) however does not hold, for instance Fig. 5 shows that $X = NENEN$ is down-compatible with a word Y' , while it is not up-compatible with the word Y – obtained from Y' by changing all S steps with N steps – and there is no word Y which is up-compatible with X .

5 Further Work

Using an exhaustive program written in SAGE we were led to formulate some conjectures, tested for $|X| \leq 13$. In a polyomino, the properties of being both a pseudo square and a permutomino determine strong symmetries of the word X , more precisely it seems that

Conjecture 1. If the word X of the form $N^+(E \vee N)^*N^+$ is exact, then it is a palindrome having odd length.

The reader can test this conjecture on all the examples presented throughout the paper. Assuming Conjecture 1, by symmetry, we have that each word Y related to X in the decomposition of P as pseudo-square is a palindrome of odd length too. The converse of Conjecture 1 does not hold, as one can easily check with the word $X = NNNENNN$. Our main goal would be to characterize the palindromic words which determine *psc*-permutominoes and then possibly

enumerate these words according to the length. For instance, we have three exact words of length 5, $NNENN$ (both in the up-growing and in the down-growing directions), $NEEEN$ and $NENEN$ (only in the down-growing direction).

Another challenging and more general problem concerns the study of pseudo-hexagon convex permutominoes; in this case, as established in [1], many of the basic properties of pseudo-square convex polyominoes do not hold so we do not have unicity of the decomposition. Also the problem of studying non convex permutominoes which tile the plane by translation is completely open.

References

1. Brlek, S., Frosini, A., Rinaldi, S., Vuillon, L.: Tilings by translation: enumeration by a rational language approach. *The Electronic Journal of Combinatorics* 13(1) (2006)
2. Bernini, A., Disanto, F., Pinzani, R., Rinaldi, S.: Permutations defining convex permutominoes. *Journal of Integer Sequences, J. Int. Seq.* 10, Article 07.9.7 (2007)
3. Disanto, F., Frosini, A., Pinzani, R., Rinaldi, S.: A closed formula for the number of convex permutominoes. *El. J. Combinatorics* 14, #R57 (2007)
4. Fraenkel, A.S.: The Rat game and the Mouse game, to appear in *Games of No Chance 4*
5. Girault-Beauquier, D., Nivat, M.: Tiling the plane with one tile. In: *Proceedings of the Sixth Annual Symposium on Computational Geometry, Berkley, California, United States, June 7-9 (1990)*
6. Golomb, S.W.: *Polyominoes: Puzzles, Patterns, Problems, and Packings*. Princeton Academic Press, London (1996)
7. Kassel, C., Lascoux, A., Reutenauer, C.: The singular locus of a Schubert variety. *J. Algebra* 269, 74–108 (2003)
8. Wijshoff, H.A.G., Van Leeuwen, J.: Arbitrary versus periodic storage schemes and tessellations of the plane using one type of polyomino. *Inform. Control* 62, 1–25 (1991)

Characterization of $\{-1, 0, +1\}$ Valued Functions in Discrete Tomography under Sets of Four Directions

Sara Brunetti¹, Paolo Dulio², and Carla Peri³

¹ Dipartimento di Scienze Matematiche e Informatiche, Università di Siena,
Pian dei Mantellini 44, 53100, Siena, Italy

sara.brunetti@unisi.it

² Dipartimento di Matematica “F. Brioschi”, Politecnico di Milano,
Piazza Leonardo da Vinci 32, I-20133 Milano, Italy

paolo.dulio@polimi.it

³ Università Cattolica S. C.,
Via Emilia Parmense, 84 29122 Piacenza, Italy

carla.peri@unicatt.it

Abstract. In this paper we use the algebraic approach to Discrete Tomography introduced by Hajdu and Tijdeman to study functions $f : \mathbb{Z}^2 \rightarrow \{-1, 0, +1\}$ which have zero line sums along the lines corresponding to certain sets of four directions.

Keywords: Discrete Tomography, X -ray, Unique Reconstruction, Generating Function.

1 Introduction

The main problems of Discrete Tomography deal with the reconstruction and the uniqueness questions. The first problem consists in the retrieval of an unknown finite subset of points of the so called *lattice set* \mathbb{Z}^2 , from the knowledge of its X -rays taken along a given set of directions. The second problem focuses on the unique determination of a lattice set by means of its X -rays taken along a given set S of lattice directions. In both cases we have to do with lattice sets (see [3] and [4] for an overview of the topics). Recently, Hajdu and Tijdeman introduced in [1] an algebraic approach for discrete tomographical problems which is based on generating functions and divisibility properties of polynomials. This reveals to be useful for studying both the above problems, and especially uniqueness. In this paper we follow this idea in order to analyze functions $f : \mathbb{Z}^2 \rightarrow \{-1, 0, +1\}$ which has zero line sums along the lines corresponding to certain sets of four directions. Our study begins from the following result in [2]: for any fixed rectangle A in \mathbb{Z}^2 there exists a “valid” set S of four directions (at least when A is not too “small“), depending only on the size of A , such that any two subsets of A can be distinguished by means of X -rays along the directions in S . The proof was given by constructing explicitly a valid set S of four directions in any

possible case. We focus on sets S of four directions. We first prove a necessary condition for S to be a set of directions of uniqueness (Theorem 2). Then, in Theorem 3, we find an algebraic necessary condition for a $\{-1, 0, +1\}$ valued function f , under sets of four lattice directions taken according to Theorem 2, in order to have $|f| \leq 1$. We also provide algebraic conditions for constructing such functions. Finally we deduce a uniqueness result in Corollary 1.

2 Notations and Preliminaries

For all notations we refer to [2]. Here we just recall the notion of *valid set of directions*. Let $a, b \in \mathbb{Z}$ with $\gcd(a, b) = 1$ and $a \geq 0$, with the further assumption that $b = 1$ if $a = 0$. We call (a, b) a *lattice direction*. Let $A = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$. If $f : A \rightarrow \mathbb{Z}$ is a function, then $|f| = \max_{(i,j) \in A} \{|f(i, j)|\}$. A set $S = \{(a_k, b_k)\}_{k=1}^d$ of d directions is said to be *valid* for A , if $\sum_{k=1}^d a_k < m$, and $\sum_{k=1}^d |b_k| < n$. In [2] L. Hajdu proved the following

Theorem 1. *Let m and n be integers with $m \geq n \geq 5$ and $m \neq 6$, and let $A = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$. Put $d = 5$ if $(m, n) \in \{(8, 6), (8, 8), (10, 6), (12, 6)\}$, and $d = 4$ otherwise. Then there exists a valid set S for A consisting of d directions depending only on n if $n \geq 15$, and on m and n otherwise, such that if the function $f : A \rightarrow \mathbb{Z}$ has zero line sums along the lines corresponding to the directions in S and $|f| \leq 1$, then f is identically zero.*

It follows that, under the above assumptions, if two functions $f, g : A \rightarrow \{0, 1\}$ are tomographically equivalent (that is is they have equal line sums along the lines corresponding to the directions in S), then $f = g$. When $n \leq 15$ analogous results hold, but depending also on m and in some cases with $d = 5$ directions. See [2] for details.

Let (a, b) be a lattice direction, then

$$f_{(a,b)}(x, y) = \begin{cases} x^a y^b - 1, & \text{if } a > 0, b > 0 \\ x^a - y^{-b}, & \text{if } a > 0, b < 0 \\ x - 1, & \text{if } a = 1, b = 0 \\ y - 1, & \text{if } a = 0, b = 1 \end{cases}$$

For any set S of valid lattice directions, we denote by $F_S(x, y) = \sum f(i, j)x^i y^j$ the polynomial associated to S defined as follows (see [2, pag. 19]):

$$F_S(x, y) = \prod_{(a,b) \in S} f_{(a,b)}(x, y).$$

For any function $f : A \rightarrow \mathbb{Z}$, its *generating function* is defined by

$$G_f(x, y) = \sum_{(i,j) \in A} f(i, j)x^i y^j.$$

If f has zero line sums along the lines taken in the directions of S , then $F_S(x, y)$ divides $G_f(x, y)$ over \mathbb{Z} ([1, Lemma 3.1]).

Further, we say that a point $(i, j) \in A$ is a *multiple positive point* for f (or G_f) if $f(i, j) > 1$. Analogously, $(i, j) \in A$ is said to be a *multiple negative point* for f if $f(i, j) < -1$.

3 Necessary Conditions for Unique Reconstructions

In view of the tomographic problem of recognizing an arbitrary set of points in \mathbb{Z}^2 by means of X -rays corresponding to a given set S of directions, an easy general necessary condition is the following.

Lemma 1. *Let S be any valid set of lattice directions, and let $F_S(x, y) = \sum f(i, j)x^i y^j$ be the polynomial associated to S . If $f(i, j) \in \{-1, 0, 1\}$, then there exist two tomographically equivalent lattice sets with respect to S .*

Proof. Let F_1, F_2 be the sets of points (i, j) such that $f(i, j) = -1$ and $f(i, j) = 1$, respectively. Then $F_S(x, y)$ has zero sums along the directions in S , so that F_1, F_2 are tomographically equivalent. \square

The following is a useful result proved by L. Hajdu [2, Lemma 3.2], related to the problem of unique reconstruction.

Lemma 2. *Let $S = \{(a_k, b_k)\}_{k=1}^d$ be any valid set of lattice directions. Suppose that $F_S(x, y) = \sum f(i, j)x^i y^j$ has a coefficient $f(i, j)$ outside the set $\{-1, 0, 1\}$. Then there exist two disjoint subsets S_1 and S_2 of S , such that $|S_1| = |S_2| \pmod{2}$ and*

$$\sum_{(a,b) \in S_1} (a, b) = \sum_{(a,b) \in S_2} (a, b) \tag{1}$$

Consider now a set $S = \{u_1, u_2, u_3, u_4\}$ of four distinct lattice directions $u_1 = (a, p), u_2 = (b, q), u_3 = (c, r), u_4 = (d, s)$. In this case, Lemma 3 in [2] can be improved.

Lemma 3. *Let $S = \{u_1, u_2, u_3, u_4\}$, and $F_S(x, y) = \sum f(i, j)x^i y^j$ its associated polynomial. Then $F_S(x, y)$ has a coefficient $f(i, j)$ outside the set $\{-1, 0, 1\}$ if and only if there exist two disjoint subsets S_1 and S_2 of S , such that $|S_1| = |S_2| \pmod{2}$ and*

$$\sum_{(a,b) \in S_1} (a, b) = \sum_{(a,b) \in S_2} (a, b) \tag{2}$$

Proof. If $F_S(x, y)$ has a coefficient $f(i, j) \notin \{-1, 0, 1\}$, then, by [2, Lemma 3] there exist two disjoint subsets S_1 and S_2 of S , such that $|S_1| = |S_2| \pmod{2}$ and (2) holds.

Suppose now the contrary. Since $|S| = 4$, two disjoint subsets S_1 and S_2 of S can exist, such that $|S_1| = |S_2| \pmod{2}$ and (2) holds, if and only if one direction is the sum of the remaining three, or the sum of one pair equals the sum of the other. Therefore, up to permute the order in the choice of the directions, we must consider the following two cases.

CASE 1. $u_4 = u_1 + u_2 + u_3$

CASE 2. $u_4 = u_1 + u_2 - u_3$

Let's show that in both cases we can always find a coefficient $f(i, j)$ which does not belong to the set $\{-1, 0, 1\}$.

CASE 1. Let $u_4 = u_1 + u_2 + u_3$ so that $(d, s) = (a + b + c, p + q + r)$. Since S consists of distinct directions, without loss of generality we can assume

$$0 \leq a \leq b \leq c, \quad b, c \neq 0 \tag{3}$$

Moreover, $F_S(x, y)$ is formed by monomials as in Table 1 (up to a common factor y^h if p, q, r are not all positive). Consequently the coefficient 2 vanishes if the monomial $x^{a+b+c}y^{p+q+r}$ is equal to one of the monomials with a positive coefficient. Note that

1. $x^{2(a+b+c)}y^{2(p+q+r)} \neq x^{a+b+c}y^{p+q+r}$, by (3).
2. If $x^{2a+b+c}y^{2p+q+r} = x^{a+b+c}y^{p+q+r}$, then $a = 0$ and $p = 0$, which is impossible for a direction (a, p) . For the same reason it is $x^{b+c}y^{q+r} \neq x^{a+b+c}y^{p+q+r}$
3. $x^{a+2b+c}y^{p+2q+r} \neq x^{a+b+c}y^{p+q+r}$ since $b \neq 0$. For the same reason it is $x^{a+c}y^{p+r} \neq x^{a+b+c}y^{p+q+r}$
4. $x^{a+b+2c}y^{p+q+2r} \neq x^{a+b+c}y^{p+q+r}$ since $c \neq 0$.
5. $x^{a+b}y^{p+q} \neq x^{a+b+c}y^{p+q+r}$ since $c \neq 0$.
6. $1 \neq x^{a+b+c}y^{p+q+r}$ since $a + b + c > 0$.

Table 1. Monomials of $F_S(x, y)$ in CASE 1

sign +	sign -
$x^{2(a+b+c)}y^{2(p+q+r)}$	$x^{2a+2b+c}y^{2p+2q+r}$
$x^{2a+b+c}y^{2p+q+r}$	$x^{2a+b+2c}y^{2p+q+2r}$
$x^{a+2b+c}y^{p+2q+r}$	$x^{a+2b+2c}y^{p+2q+2r}$
$x^{a+b+2c}y^{p+q+2r}$	$2x^{a+b+c}y^{p+q+r}$
$x^{a+b}y^{p+q}$	$x^a y^p$
$x^{a+c}y^{p+r}$	$x^b y^q$
$x^{b+c}y^{q+r}$	$x^c y^r$
1	

This shows that in CASE 1 there is a coefficient $f(i, j)$ which does not belong to the set $\{-1, 0, 1\}$.

CASE 2. Let $u_4 = u_1 + u_2 - u_3$ so that $(d, s) = (a + b - c, p + q - r)$. Without loss of generality we can assume

$$0 \leq a \leq b, \quad 0 \leq a + b - c, \quad b \neq 0 \tag{4}$$

Now $F_S(x, y)$ is formed by monomials as in Table 2 (still up to a common factor y^h if p, q, r are not all positive).

Table 2. Monomials of $F_S(x, y)$ in CASE 2

sign +	sign -
$x^{2(a+b)}y^{2(p+q)}$	$x^{2(a+b)-c}y^{2(p+q)-r}$
$x^{2a+b-c}y^{2p+q-r}$	$x^{2a+b}y^{2p+q}$
$x^{a+2b-c}y^{p+2q-r}$	$x^{a+2b}y^{p+2q}$
$2x^{a+b}y^{p+q}$	$x^{a+b+c}y^{p+q+r}$
$x^{a+c}y^{p+r}$	$x^{a+b-c}y^{p+q-r}$
$x^{b+c}y^{q+r}$	$x^a y^p$
1	$x^b y^q$
	$x^c y^r$

The coefficient 2 vanishes if the monomial $x^{a+b}y^{p+q}$ equals one with negative sign. Note that

1. If $x^{2(a+b)-c}y^{2(p+q)-r} = x^{a+b}y^{p+q}$, then $c = a + b$ and $r = p + q$, deriving $d = 0$ and $s = 0$, which is impossible for a direction (d, s) . For the same reason it is $x^c y^r \neq x^{a+b}y^{p+q}$
2. If $x^{2a+b}y^{2p+q} = x^{a+b}y^{p+q}$, then $a = 0$ and $p = 0$, which is impossible for a direction (a, p) . For the same reason it is $x^b y^q \neq x^{a+b}y^{p+q}$
3. $x^{a+2b}y^{p+2q} \neq x^{a+b}y^{p+q}$, since $b \neq 0$ by (4).
4. If $x^{a+b+c}y^{p+q+r} = x^{a+b}y^{p+q}$, then $c = 0$ and $r = 0$, which is impossible for a direction (c, r) . Analogously we have $x^{a+b-c}y^{p+q-r} \neq x^{a+b}y^{p+q}$.
5. $x^a y^p \neq x^{a+b}y^{p+q}$, since $b \neq 0$ by (4).

Consequently, also in CASE 2 there is a coefficient $f(i, j)$ which does not belong to the set $\{-1, 0, 1\}$, and the proof is complete. □

From Lemma 3 we derive the following necessary condition of unique reconstruction.

Theorem 2. *The sets of four directions that uniquely determine the lattice sets in a finite grid must be of the form $\{u_1, u_2, u_3, u_1 + u_2 \pm u_3\}$.*

Proof. By Lemma 1 a set $S = \{u_1, u_2, u_3, u_4\}$ of four direction could be a set of uniqueness if the associated polynomial $F_S(x, y) = \sum f(i, j)x^i y^j$ has at least a coefficient $f(i, j) \notin \{-1, 0, 1\}$. By CASE 1 and CASE 2 in Lemma 3 this occurs if and only if $u_4 = u_1 + u_2 \pm u_3$. □

Remark 1. Let S be a set of four lattice directions such that $F_S(x, y)$ has some coefficient outside $\{-1, 0, 1\}$. By Lemma 3 we know that there exist two disjoint subsets S_1, S_2 of S , such that $|S_1| = |S_2| \pmod{2}$, and $\sum_{u \in S_1} u = \sum_{u \in S_2} u$, and by Theorem 2 we get that $S = S_1 \cup S_2$, where $S_1 = \{u_1, u_2, u_3\}, S_2 = \{u_4\}$ in CASE 1, and $S_1 = \{u_1, u_2\}, S_2 = \{u_3, u_4\}$ in CASE 2. For such a set S we define the set $S_1 - S_2$ as follows

$$S_1 - S_2 = \{\pm(u_i - u_j), u_i \in S_1, u_j \in S_2\}$$

4 Sets of Four Directions for Unique Reconstructions

In this section we analyze the functions $g : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ which have zero line sums along the lines corresponding to a given set of directions S , $|g| \leq 1$, and vanish outside a finite lattice set. Notice that these properties are invariant by integer translations, so that we can argue up to integer translations, when necessary.

Let $S = \{u_1, u_2, u_3, u_4\}$ be a valid set of four lattice directions, and let

$$F_S(x, y) = \prod_{(a,b) \in S} f_{(a,b)}(x, y) = \sum f(i, j) x^i y^j$$

be its associated polynomial. Define $Q(S) = \{(i, j) \in \mathbb{Z}^2 : f(i, j) \neq 0\}$. We partition the set $Q(S)$ into two disjoint subsets, according to the sign of the *weight* $f(i, j)$ of the point $(i, j) \in Q(S)$:

$$\mathcal{P} = \{(i, j) \in Q(S) : f(i, j) > 0\} \quad \mathcal{N} = \{(i, j) \in Q(S) : f(i, j) < 0\} \tag{5}$$

Notice that if $b \geq 0$ for all $(a, b) \in S$ then the set \mathcal{P} consists of the points

$$\begin{aligned} & \mathbf{0}, & u_1 + u_2, & u_1 + u_3, & u_1 + u_4, \\ & u_2 + u_3, & u_2 + u_4, & u_3 + u_4, & u_1 + u_2 + u_3 + u_4 \end{aligned} \tag{6}$$

and the set \mathcal{N} consists of the points

$$\begin{aligned} & u_1, & u_2, & u_3, & u_4, \\ & u_1 + u_2 + u_3, & u_1 + u_2 + u_4, & u_1 + u_3 + u_4, & u_2 + u_3 + u_4. \end{aligned} \tag{7}$$

not all necessarily distinct. If $b < 0$ for some $(a, b) \in S$, then the sets \mathcal{P} and \mathcal{N} consist of the points (6) and (7), respectively, translated by the vector $(0, -h)$, where h is the sum of negative values of b for $(a, b) \in S$. In the following we shall assume that \mathcal{P} and \mathcal{N} are represented by (6) and (7), respectively, since the properties we are looking for are invariant by integer translations. From the algebraic viewpoint, this corresponds with substituting the polynomial $F_S(x, y)$ by the rational function $y^d F_S(x, y)$.

For $u = (h, k) \in \mathbb{Z}^2$, let $f_-^u : \mathbb{Z}^2 \rightarrow \mathbb{Z}$, $f_+^u : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ be the maps whose generating functions are $G_{f_-^u}(x, y) = (x^h y^k - 1)F_S(x, y)$ and $G_{f_+^u}(x, y) = (x^h y^k + 1)F_S(x, y)$, respectively. Notice that for $h < 0$ or $k < 0$, $G_{f_-^u}, G_{f_+^u}$ are rational functions, which can be mapped to polynomials by integer translations.

Lemma 4. *Let $S = \{u_1, u_2, u_3, u_1 + u_2 \pm u_3\}$ be a set of four lattice directions, and let $u \in \mathbb{Z}^2$. If $u \in S$ or $-u \in S$, then $|f_-^u| > 1$.*

Proof. If $u = (h, k) \in S$, then $u \in \mathcal{N}$ by (7), so that $f(h, k) < 0$. By definition the coefficient of the monomial $x^h y^k$ in the generating function $G_{f_-^u}$ of f_-^u is given by $f(0, 0) - f(h, k)$. Moreover, since $(0, 0) \in \mathcal{P}$ we have $f(0, 0) - f(h, k) \geq 2$, so that $|f_-^u| > 1$. If $-u \in S$, then $-u \in \mathcal{N}$ by (7), so that $f(-h, -k) < 0$. By definition the coefficient of the monomial of degree zero in the generating function $G_{f_-^u}$ of f_-^u is given by $f(-h, -k) - f(0, 0) \leq -2$, so that $|f_-^u| > 1$. □

Lemma 5. *Let $S = \{u_1, u_2, u_3, u_1 + u_2 \pm u_3\}$ be a set of four lattice directions. For each $u_i \in S$ there exists $u = (h, k) \in (S_1 - S_2)$ such that u_i is a multiple negative point for f_+^u .*

Proof. Let $u_i = (h_i, k_i) \in S$ and let $u = (h, k) \in (S_1 - S_2)$ such that $u = u_i - u_j$ for some $u_j = (h_j, k_j)$. Then $(h_i, k_i) = (h + h_j, k + k_j)$ so that the coefficient of the monomial $x^{h_i}y^{k_i}$ in the generating function $G_{f_+^u}$ of f_+^u is given by $f(h_i, k_i) + f(h_j, k_j)$. Since $u_i, u_j \in \mathcal{N}$, then $f(h_i, k_i) + f(h_j, k_j) \leq -2$. Therefore, u_i is a multiple negative point for f_+^u . \square

Lemma 6. *Let $S = \{u_1, u_2, u_3, u_4\}$ be a set of four lattice directions, such that $u_4 = u_1 + u_2 \pm u_3$. Then the following hold*

1. $|f_-^u| \leq 1$ if and only if $u = 0$, or $u \in (S_1 - S_2)$ and $u_4 = u_1 + u_2 + u_3$, or $u \in (S_1 - S_2) \cup \{\pm(u_1 + u_2)\}$ and $u_4 = u_1 + u_2 - u_3$.
2. $|f_+^u| \leq 1$ if and only if $u \in S$ or $-u \in S$.

Proof. We first note that if $u = 0$ then $|f_-^0|$ is identically zero, so that $|f_-^0| \leq 1$, while $|f_+^0| = 2f$, so that $|f_+^0| > 1$.

If $u \in S$ or $-u \in S$, then, by Lemma 4 $|f_-^u| > 1$. Consequently, in order to get $|f_-^u| \leq 1$, we must choose $\pm u \notin S$. Analogously, if $u \in (S_1 - S_2)$, then $u = \pm(u_i - u_j)$ for some $u_i, u_j \in S$. By Lemma 5, either u_i or u_j is a multiple negative point of f_+^u so that $|f_+^u| > 1$. Consequently, in order to get $|f_+^u| \leq 1$, we must choose $u \notin (S_1 - S_2)$.

Consider the multiple (positive or negative) point $w = (1/2)(u_1 + u_2 + u_3 + u_4) \in Q(S)$ for F_S (see Table 1 and Table 2). Let $z \in \mathbb{Z}^2$ such that $z = w + u$. If $z \notin Q(S)$ then $f_\pm^u(z) = f(w)$, so that z is a multiple point (positive or negative) for f_\pm^u , and consequently $|f_\pm^u| > 1$. Then, in order to get $|f_\pm^u| \leq 1$, we need $z \in Q(S)$.

Let us first suppose that $u_4 = u_1 + u_2 + u_3$, so that w is a multiple negative point for F_S . Then, by (6) and (7), to have $z \in Q(S)$, the vector u must be selected as follows

$$\begin{aligned} u = \mathbf{0}, \quad u = \pm u_1, \quad u = \pm u_2, \quad u = \pm u_3, \\ u = \pm u_4, u = \pm(u_4 - u_1), u = \pm(u_4 - u_2), u = \pm(u_4 - u_3), \end{aligned}$$

The case $u = \mathbf{0}$ has already been considered. By Lemma 4 the cases $\pm u \in S$ cannot give $|f_-^u| \leq 1$. To this, the only possible choice is $u \in (S_1 - S_2)$. The explicit computations of $G_{f_-^u}(x, y) = (x^h y^k - 1)F_S(x, y)$ for $(h, k) \in (S_1 - S_2)$ show that in fact $|f_-^u| \leq 1$ in all the cases, and (1) is proved.

The cases $u \in \{\pm(u_4 - u_1), \pm(u_4 - u_2), \pm(u_4 - u_3)\}$ correspond to $u \in (S_1 - S_2)$, and consequently cannot give $|f_+^u| \leq 1$ by Lemma 5. To this, the only possible choices are $u \in S$ or $-u \in S$. The explicit computations of $G_{f_+^u}(x, y) = (x^h y^k + 1)F_S(x, y)$ for such h, k show that in fact $|f_+^u| \leq 1$ in all the cases, and (2) is proved.

Now, assume $u_4 = u_1 + u_2 - u_3$. Then, by (6) and (7), to have $z \in Q(S)$ the vector u must be selected as follows

$$\begin{aligned} u = \mathbf{0}, \quad u = \pm u_1, \quad u = \pm u_2, \quad u = \pm u_3 \\ u = \pm u_4, \quad u = \pm(u_3 - u_1), \quad u = \pm(u_3 - u_2), \quad u = \pm(u_1 + u_2) \end{aligned}$$

The case $u = \mathbf{0}$ has already been considered above. If $u \in S$ or $-u \in S$ then $|f^u| > 1$, by Lemma 4. Therefore, we remain with $u \in (S_1 - S_2) \cup \{\pm(u_1 + u_2)\}$, and the explicit computations of $G_{f^u}(x, y) = (x^h y^k - 1)F_S(x, y)$ for such u show that it is always $|f^u| \leq 1$. Therefore, (1) is proved also in this case.

Differently, to get $|f^u_+| \leq 1$, we must avoid $u \in (S_1 - S_2)$, by Lemma 5, and $u = \pm(u_1 + u_2)$, as it is shown by the explicit computations of $G_{f^u_+}(x, y) = (x^h y^k + 1)F_S(x, y)$. Thus, we remain with $u \in S$ or $-u \in S$. The explicit computations of $G_{f^u_+}(x, y) = (x^h y^k + 1)F_S(x, y)$ for such u show that it is always $|f^u_+| \leq 1$. Therefore (2) is proved also in this case. \square

From the above lemmas, we can derive a necessary condition for a $\{-1, 0, +1\}$ valued function g , which has zero line sums along the lines corresponding to sets of four lattice directions taken according to Theorem 2, in order to have $|g| \leq 1$.

Theorem 3. *Let $S = \{u_1, u_2, u_3, u_1 + u_2 \pm u_3\}$ be a set of four lattice directions. Let $g : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ be a non trivial function which has zero line sums along the lines corresponding to the directions in S , and vanishes outside a finite lattice set. If $|g| \leq 1$ then there exists $r \in \mathbb{N}$ such that*

$$G_g(x, y) = \sum_{t=1}^r \delta(t)x^{i(t)}y^{j(t)}F_S(x, y), \tag{8}$$

where $\delta(t) = \pm 1$, and for each $t \in \{1, \dots, r\}$, there exists $t' \in \{1, \dots, r\}$ such that the vector $u(t) = (i(t), j(t)) - (i(t'), j(t'))$ satisfies the following conditions

1. $u(t) = 0$, or $u(t) \in (S_1 - S_2)$ and $u_4 = u_1 + u_2 + u_3$, or $u(t) \in (S_1 - S_2) \cup \{\pm(u_1 + u_2)\}$ and $u_4 = u_1 + u_2 - u_3$, if $\delta(t) \neq \delta(t')$.
2. $u(t) \in S$ or $-u(t) \in S$ if $\delta(t) = \delta(t')$.

Proof. By Theorem 1 in [1], $G_g(x, y) = P(x, y)F_S(x, y)$ for some polynomial $P(x, y) \in \mathbb{Z}[x, y]$. Each monomial $a_{ij}x^i y^j$ of $P(x, y)$ can be split into a sum of monomials with coefficients ± 1 , so that $P(x, y) = \sum_{t=1}^r \delta(t)x^{i(t)}y^{j(t)}$, and

$$G_g(x, y) = \sum_{t=1}^r \delta(t)x^{i(t)}y^{j(t)}F_S(x, y),$$

where $\delta(t) = \pm 1$. Consider the multiple (positive or negative) point

$$w = (1/2)(u_1 + u_2 + u_3 + u_4)$$

for F_S . We have $|f(w)| = 2$, where f is the function with generating function F_S . By multiplying $F_S(x, y)$ times $x^{i(t)}y^{j(t)}$, the value $f(w) = 2$ of the

the multiple (positive or negative) point w is attached to the translated point $w + (i(t), j(t))$. Since $|g| \leq 1$, such value must be reduced by adding to the corresponding monomial a monomial with same degree and coefficient with opposite sign. Thus, for each $t \in \{1, \dots, r\}$, some point $z_t \in Q(S)$ must exist, and some monomial $\delta(t')x^{i(t')}y^{j(t')} \neq \delta(t)x^{i(t)}y^{j(t)}$ of $P(x, y)$, such that $z_t + (i(t'), j(t')) = w + (i(t), j(t))$. It follows that the function

$$\left(\delta(t)x^{i(t)}y^{j(t)} + \delta(t')x^{i(t')}y^{j(t')} \right) F_S(x, y) = x^{i(t')}y^{j(t')} \left(\delta(t)x^{i(t)-i(t')}y^{j(t)-j(t')} + \delta(t') \right) F_S(x, y)$$

has no multiple points. Let us define $u(t) = (i(t), j(t)) - (i(t'), j(t')) = z_t - w$. By Lemma 6 applied to the vector $u(t)$ and to the function

$$\left(\delta(t)x^{i(t)-i(t')}y^{j(t)-j(t')} + \delta(t') \right) F_S(x, y),$$

we get either (I) or (II), depending on the signs of $\delta(t)$ and $\delta(t')$, which provide a necessary condition for $|g| \leq 1$. □

We now provide some algebraic conditions which show how a $\{-1, 0, +1\}$ valued function g , with required norm $|g| \leq 1$, can be constructed. We first fix some notations. Let $g : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ be a non trivial function whose generating function is defined by $G_g(x, y) = P(x, y)F_S(x, y)$, for some polynomial $P(x, y)$, consisting of r monomials. For $h \in \{1, \dots, r\}$, let $g_h : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ be the function whose generating function is determined by multiplying $F_S(x, y)$ time the first h monomials of $P(x, y)$. Let

$$Q_h(S) = \{(i, j) \in \mathbb{Z}^2 : g_h(i, j) \neq 0\}$$

and

$$\mathcal{P}_h = \{(i, j) \in Q_h(S) : g_h(i, j) > 0\}, \quad \mathcal{N}_h = \{(i, j) \in Q_h(S) : g_h(i, j) < 0\}.$$

In particular $Q_0(S) = Q(S)$, $\mathcal{P}_0 = \mathcal{P}$, and $\mathcal{N}_0 = \mathcal{N}$. Let M_h denote the set of multiple points of $Q_h(S)$, and let M_h^+, M_h^- be the subsets of M_h formed by the points of M_h belonging to \mathcal{P}_h and \mathcal{N}_h , respectively. In particular $M_0 = \{w\}$, namely the multiple point of $Q(S)$. Moreover, for any pair of sets X, Y , we define

$$X - Y = \{\pm(u - v), u \in X, v \in Y\}.$$

Proposition 1. *Let $S = \{u_1, u_2, u_3, u_1 + u_2 \pm u_3\}$ be a set of four lattice directions. Let $g : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ be a non trivial function whose generating function is defined by $G_g(x, y) = \sum_{t=1}^r \delta(t)x^{i(t)}y^{j(t)}F_S(x, y)$, where $\delta(t) = \pm 1$. For each $h \in \{1, \dots, r - 1\}$, consider the monomial $\delta(h + 1)x^{i(h+1)}y^{j(h+1)}$. Suppose that $\delta(h + 1) = 1$ and the following conditions hold*

1. *If $(c, d) \in M_h^+$ then $(i(h + 1), j(h + 1)) = (c, d) - (e, f)$, with $(e, f) \in \mathcal{N}$*
2. *If $(c, d) \in M_h^-$ then $(i(h + 1), j(h + 1)) = (c, d) - (e, f)$, with $(e, f) \in \mathcal{P}$*

- 3. $(i(h + 1), i(h + 1)) \notin (\mathcal{P}_h - \mathcal{P}) \cup (\mathcal{N}_h - \mathcal{N})$.
- 4. $(i(h + 1), i(h + 1)) + w \in \mathcal{N}_h$ if $w \in M_0^+$, $(i(h + 1), i(h + 1)) + w \in \mathcal{P}_h$ if $w \in M_0^-$

Then $M_{h+1} \subseteq M_h$.

Proof. We give the proof just in the case when both w and (c, d) are positive. The other cases can be obtained by changing the argument in obvious ways. If $(c, d) \in M_h^+$ we add a monomial $x^{i(h+1)}y^{i(h+1)}$ with $(i(h + 1), i(h + 1)) = (c, d) - (e, f)$, $(e, f) \in \mathcal{N}$ and $(i(h + 1), i(h + 1)) \notin (\mathcal{P}_h - \mathcal{P}) \cup (\mathcal{N}_h - \mathcal{N})$. This ensures that the negative point (e, f) overlaps (c, d) , and consequently that the multiplicity of (c, d) is lowered. Moreover, in order to avoid new multiple points in the updated configuration, we need $(i(h + 1), i(h + 1)) \notin (\mathcal{P}_h - \mathcal{P}) \cup (\mathcal{N}_h - \mathcal{N})$, and $(i(h + 1), i(h + 1)) + w \in \mathcal{N}_h$. □

Remark 2. By the above proposition we can construct a $\{-1, 0, +1\}$ valued function $g : \mathbb{Z}^2 \rightarrow \mathbb{Z}$, with required norm $|g| \leq 1$, starting from a given polynomial $P(x, y)F_S(x, y)$ which associated configuration has multiple points. To this we add progressively new monomials according to Proposition 1. See the following example, first part. Of course, we can also work with monomials with negative sign, up to exchange \mathcal{P} with \mathcal{N} , and \mathcal{P}_h with \mathcal{N}_h in the above conditions. See for instance the second part in the following example.

Example 1. Consider the following set of four directions

$$S = \{(1, 0), (0, 1), (1, 2), (2, 1)\},$$

and the polynomial associated to S :

$$F_S(x, y) = \sum f(i, j)x^i y^j = (x - 1) \cdot (y - 1) \cdot (x^2 \cdot y - 1) \cdot (x \cdot y^2 - 1)$$

Let $g_h : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ with generating function given by

$$\begin{aligned} G_{g_h}(x, y) &= (x \cdot y + x^2 y^2 + x + x^2 y) \cdot (x - 1) \cdot (y - 1) \cdot (x^2 \cdot y - 1) \cdot (x \cdot y^2 - 1) = \\ &= x^6 \cdot y^6 - x^6 \cdot y^4 - \boxed{x^5 \cdot y^6} + x^5 \cdot y^5 - x^5 \cdot y^3 + x^5 \cdot y^2 - \boxed{2 \cdot x^4 \cdot y^5} + x^4 \cdot y^4 + x^4 \cdot y^3 \\ &- x^4 \cdot y^2 + x^4 \cdot y + x^3 \cdot y^5 - x^3 \cdot y^4 + x^3 \cdot y^3 + x^3 \cdot y^2 - \boxed{2 \cdot x^3 \cdot y} + x^2 \cdot y^4 - x^2 \cdot y^3 + \\ &x^2 \cdot y - x^2 - x \cdot y^2 + x \end{aligned}$$

Notice that the multiple negative point $(4, 5)$ can be mapped to a simple point by adding the monomial $x^3 y^4$, which corresponds to the choice $(e, f) = (1, 1)$. This satisfies the condition $(3, 4) \notin (\mathcal{P}_h - \mathcal{P}) \cup (\mathcal{N}_h - \mathcal{N})$ (since $(i, j) + (3, 4) \notin \mathcal{P}_h$ for each $(i, j) \in \mathcal{P}$, and $(i, j) + (3, 4) \notin \mathcal{N}_h$ for each $(i, j) \in \mathcal{N}$). Moreover $(3, 4) + (2, 2) = (5, 6) \in \mathcal{N}_h$. Therefore, all conditions in Proposition 1 are fulfilled, and consequently the addition of $x^3 y^4$ does not introduce new multiple points in the updated configuration. Indeed we get

$$\begin{aligned}
 G_{g_{h+1}}(x, y) &= (x \cdot y + x^2 y^2 + x + x^2 y + x^3 y^4) \cdot (x - 1) \cdot (y - 1) \cdot (x^2 \cdot y - 1) \cdot (x \cdot y^2 - 1) \\
 &= x^7 \cdot y^8 - x^7 \cdot y^7 - x^6 \cdot y^8 + x^6 \cdot y^7 + x^6 \cdot y^5 - x^6 \cdot y^4 - x^5 \cdot y^7 + x^5 \cdot y^6 - \boxed{x^5 \cdot y^3} \\
 &+ x^5 \cdot y^2 + x^4 \cdot y^7 - x^4 \cdot y^6 - x^4 \cdot y^5 + x^4 \cdot y^3 - x^4 \cdot y^2 + x^4 \cdot y + x^3 \cdot y^3 + x^3 \cdot y^2 \\
 &- \boxed{2 \cdot x^3 \cdot y} + x^2 \cdot y^4 - x^2 \cdot y^3 + x^2 \cdot y - x^2 - x \cdot y^2 + x,
 \end{aligned}$$

so that the number of multiple points decreases.

The multiple point which remains can be eliminated by adding the monomial $x^3 y$, which corresponds to the choice $(e, f) = (0, 0)$. In fact, we have $(i, j) + (3, 1) \notin (\mathcal{P}_{h+1} - \mathcal{P}) \cup (\mathcal{N}_{h+1} - \mathcal{N})$, and $(3, 1) + (2, 2) = (5, 3) \in \mathcal{N}_{h+1}$ as before. Therefore, since no new multiple point appears, we get

$$\begin{aligned}
 (x \cdot y + x^2 y^2 + x + x^2 y + x^3 y^4 + x^3 y) \cdot (x - 1) \cdot (y - 1) \cdot (x^2 \cdot y - 1) \cdot (x \cdot y^2 - 1) &= \\
 x^7 \cdot y^8 - x^7 \cdot y^7 + x^7 \cdot y^5 - x^7 \cdot y^4 - x^6 \cdot y^8 + x^6 \cdot y^7 - x^6 \cdot y^3 + x^6 \cdot y^2 - x^5 \cdot y^7 + x^5 \cdot y^6 &- \\
 - x^5 \cdot y^4 + x^5 \cdot y^3 + x^4 \cdot y^7 - x^4 \cdot y^6 - x^4 \cdot y^5 + x^4 \cdot y^4 + x^3 \cdot y^3 - x^3 \cdot y + x^2 \cdot y^4 - x^2 \cdot y^3 &+ \\
 + x^2 \cdot y - x^2 - x \cdot y^2 + x, &
 \end{aligned}$$

which provides a function $g : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ with $|g| \leq 1$, as required.

We can also add monomials with negative coefficients, by exchanging \mathcal{P} with \mathcal{N} , and \mathcal{P}_h for \mathcal{N}_h in the above conditions. For example, if we start with the polynomial $G_{g_h}(x, y) = (x \cdot y + x^2 \cdot y^2 + x + x^2 \cdot y) \cdot (x - 1) \cdot (y - 1) \cdot (x^2 \cdot y - 1) \cdot (x \cdot y^2 - 1)$ we can continue in the following way.

To eliminate the multiple negative point $-2x^4 y^5$ we can subtract the monomial $x^4 y^4$, which corresponds to the choice $(e, f) = (1, 0) \in \mathcal{N}$, consistent with the conditions $(i, j) + (4, 4) \notin \mathcal{N}_h$ for each $(i, j) \in \mathcal{P}$, and $(i, j) + (4, 4) \notin \mathcal{P}_h$ for each $(i, j) \in \mathcal{N}$. Thus we get

$$\begin{aligned}
 (x \cdot y + x^2 \cdot y^2 + x + x^2 \cdot y - x^4 \cdot y^4) \cdot (x - 1) \cdot (y - 1) \cdot (x^2 \cdot y - 1) \cdot (x \cdot y^2 - 1) &= \\
 = -x^8 \cdot y^8 + x^8 \cdot y^7 + x^7 \cdot y^8 - x^7 \cdot y^7 + x^7 \cdot y^6 - x^7 \cdot y^5 + x^6 \cdot y^7 - x^6 \cdot y^6 + x^6 \cdot y^5 - &- \\
 x^6 \cdot y^4 - x^5 \cdot y^7 + x^5 \cdot y^4 - x^5 \cdot y^3 + x^5 \cdot y^2 - x^4 \cdot y^5 + x^4 \cdot y^3 - x^4 \cdot y^2 + x^4 \cdot y + x^3 \cdot y^5 &- \\
 - x^3 \cdot y^4 + x^3 \cdot y^3 + x^3 \cdot y^2 - \boxed{2 \cdot x^3 \cdot y} + x^2 \cdot y^4 - x^2 \cdot y^3 + x^2 \cdot y - x^2 - x \cdot y^2 + x &
 \end{aligned}$$

The remaining multiple point can be eliminated by subtracting the monomial x^3 , which corresponds to the choice $(e, f) = (0, 1) \in \mathcal{N}$, consistent with the conditions $(i, j) + (3, 0) \notin \mathcal{N}_{h+1}$ for each $(i, j) \in \mathcal{P}$, and $(i, j) + (4, 4) \notin \mathcal{P}_{h+1}$ for each $(i, j) \in \mathcal{N}$. Thus we get

$$\begin{aligned}
 (x \cdot y + x^2 \cdot y^2 + x + x^2 \cdot y - x^4 \cdot y^4 - x^3) \cdot (x - 1) \cdot (y - 1) \cdot (x^2 \cdot y - 1) \cdot (x \cdot y^2 - 1) &= \\
 = -x^8 \cdot y^8 + x^8 \cdot y^7 + x^7 \cdot y^8 - x^7 \cdot y^7 + x^7 \cdot y^6 - x^7 \cdot y^5 - x^7 \cdot y^4 + x^7 \cdot y^3 + x^6 \cdot y^7 - &- \\
 x^6 \cdot y^6 + x^6 \cdot y^5 - x^6 \cdot y^3 + x^6 \cdot y^2 - x^6 \cdot y - x^5 \cdot y^7 + x^5 \cdot y^4 - x^5 \cdot y^2 + x^5 \cdot y - x^4 \cdot y^5 + &+ \\
 x^4 + x^3 \cdot y^5 - x^3 \cdot y^4 + x^3 \cdot y^3 + x^3 \cdot y^2 - x^3 \cdot y - x^3 + x^2 \cdot y^4 - x^2 \cdot y^3 + x^2 \cdot y - x^2 &- \\
 - x \cdot y^2 + x, &
 \end{aligned}$$

which provides the generating function of a function $g : \mathbb{Z}^2 \rightarrow \mathbb{Z}$ with $|g| \leq 1$, as required.

The previous results unable us to deduce the following uniqueness result.

Corollary 1. $S = \{u_1, u_2, u_3, u_1 + u_2 \pm u_3\}$ be a set of four lattice directions. Let $h, g : \mathbb{Z}^2 \rightarrow \{0, 1\}$ be tomographically equivalent with respect to S . If

$$G_{h-g}(x, y) = \sum_{t=1}^r \delta(t) x^{i(t)} y^{j(t)} F_S(x, y),$$

with $\delta(t) = \pm 1$, and the degrees $i(t), j(t)$ do not satisfy the conditions of Theorem 3, then $h = g$.

Proof. For the function $\phi = h - g$ it results $|\phi| \leq 1$. Since $G_\phi(x, y)$ does not satisfy Theorem 3 then $\phi = 0$, and consequently $h = g$. □

5 Conclusion and Perspectives

In this paper we studied $\{-1, 0, +1\}$ valued functions under certain sets of four lattice directions. From their algebraic properties we deduced a uniqueness result. A further step could consist in considering a grid A of fixed size as in the result in 2 in order to show that in fact there exist whole families of suitable valid sets of four directions depending only on the size of A , such that any two subsets of A can be distinguished by means of such sets of directions.

References

1. Hajdu, L., Tijdeman, R.: Algebraic aspects of discrete tomography. *J. Reine Angew. Math.* 534, 119–128 (2001)
2. Hajdu, L.: Unique reconstruction of bounded sets in discrete tomography. *Electron. Notes Discrete Math.* 20, 15–25 (2005)
3. Kuba, A., Herman, G.T.: *Discrete tomography*. Appl. Numer. Harmon. Anal. Birkhäuser, Boston (1999)
4. Kuba, A., Herman, G.T.: *Advances in Discrete Tomography and Its Applications*. Appl. Numer. Harmon. Anal. Birkhäuser, Boston (2007)

Growth of Discrete Projection Ghosts Created by Iteration

Imants Svalbe and Shekhar Chandra

School of Physics, Monash University, Australia

Abstract. Ghost images contain specially aligned pixels with intensities that are designed to sum to zero when projected at any of a pre-selected set of discrete angles. Ghost images find use in synthesizing the content of missing rows of image or projection space from data that contains some deliberate level of information redundancy. Here we examine the properties of ghost images that are constructed through a process of iterated convolution. An initial ghost is propagated by cumulative displacements into other discrete directions to expand the range of angles that have zero-sum projections. The discrete projection scheme used here is the finite Radon transform (FRT). We examine these accumulating ghosts to quantify the growth of their dynamic range of their pixel values and the spread of their spatial extent. After N propagations, a pair of points with intensity ± 1 can replicate to produce a maximum total intensity of 2^N . For the discrete projections of the FRT, we show that column-oriented iterations better suppress the range and rate of growth of ghost image values. After N row-based iterations, the peak pixel values of FRT ghost images grow approximately as $2^{0.8N}$. After N column-based iterations, the peak pixel values of FRT ghost images grow approximately as $2^{0.7N}$. The slower rate of expansion of pixel values for column iteration comes at the expense of fragmenting the compactness of the set of FRT projection angles that are chosen to sum to zero.

1 Introduction

Ghosts are deceptively simple spatial images composed of signed pixel values, so arranged that projections of the ghost image data vanish or sum to zero for a range of pre-selected projection angles. The addition of linear combinations of scaled and shifted patterns of ghost values to a discrete object then adds nothing to the content of some of its projected views. Reconstruction of the object from its projected views is then either not unique or has an inconsistent interpretation, depending on the projected views that are taken.

Ghosts have formed an integral part in the post-Radon history of the reconstruction of digital images from discrete projections. Katz [9] and Louis [11] showed that the fidelity of reconstructed images is predicated on the banishment of ghosts or zero projections from the input data. Ghost theory gave rise to the Katz-criterion that specifies the number of acquired 1D projected views of an object that is sufficient for accurate reconstruction of a 2D slice. Images

are uniquely and exactly reconstructable from a set of projections if and only if the set of the projection angles constrains their ghost images to be supersets of the reconstructed image dimensions.

Gardner and Gritzman [5] and Barucci et al [1] showed how polygonal arrangements of translated switching components could be used to generate ghosts at any selection of projection angles. Their work underlined the importance of these structures in fixing the sets that can be reconstructed uniquely from finite sets of projected sums. Zopf [17] examined ways to minimise the size of the arrangement of switching elements, a topic that is further addressed in Svalbe and Normand [14]. Herman and Davidi [8] examined the practical effect of adding ghosts to images reconstructed from sub-sampled sets of real projection data. The spatial extent and range of image intensities for ghosts that are created by iterated displacement of switching elements is the subject of this paper.

The discrete projection scheme that we use here is the finite Radon transform (FRT) developed by [12]. These projections are applied to data embedded on prime-sized arrays. The FRT formalism relies on the unique and exact tiling in 2D space of oriented slices in discrete images and their associated discrete Fourier transforms. This “discrete slice theorem” approach was developed by [6].

Ghosts provide a bridge between the symmetric set of discrete projections of the FRT and the more general asymmetric sets of the discrete Mojette transform [7]. The distribution of pixel grey scales in a ghost and their spatial extent in image space are crucial aspects that govern how easily they can be implemented to de-convolve mixtures of ghosts. Reconstructions that use a projection set with one or more missing projections will generate artefacts in the reconstructed image. These image artefacts can be seen to be linear combinations of ghosts, as the projected image intensities must sum to zero for each missing angle. If the image contains regions that are known to be constant, then the added mixtures of ghosts will perturb these constant values. This property suggests a way to remove image reconstruction artefacts [4].

We examine and quantify the spatial spread and the rate at which the signed pixel values in a ghost image grow under a scheme where iterative convolutions are used to increment the size of the angle set for which the discrete projections will sum to zero. Ghosts created by this method become non-minimal after just a few iteration steps. Minimal ghosts span the full extent of the image space, but contain the theoretical minimum of $2N$ pixels; N with value $+1$, and N pixels with value -1 . Methods to generate minimal ghosts are presented in [15]. Examination of the properties of minimal ghosts is presented in a companion paper [14].

The finite Radon transform is reviewed in section 2. Section 3 outlines the method of constructing ghosts by iterative convolution. Section 4 gives some examples of ghosts constructed by iterations oriented along the row and column directions. Section 5 compares quantitatively the growth patterns of these two methods. Section 6 outlines some areas where this work may be extended and summarises our current findings.

2 Discrete Projections and the Finite Radon Transform

The Finite Radon Transform (FRT) is a linear transform that preserves all of the information present in the original image. The image $I(x, y)$ is always exactly recoverable from its FRT $R(t, m)$. A fast algorithm to compute the FRT and its inverse, based on the number theoretic transform, was presented in [3]. A symmetrised form of the FRT enables a systematic redundant encoding of data makes it suitable for an error correcting internet transmission scheme [13]. Ghosts form an inherent part of the data encoding method in the latter work.

We consider data displayed on a $p \times p$ image array, where p is prime. The FRT projects this data at a set of $p + 1$ discrete projection angles. Each projection is comprised of p parallel translated rays. Each ray sums the intensity of exactly p pixels located in the image domain. The FRT assumes that the data wraps periodically at the boundaries of the image. Then $I(x + mp, y + np) = I(x, y)$ for any integers m and n . The FRT, $R(t, m)$, of an image $I(x, y)$ is defined as

$$R(t, m) = \sum I(\langle t + my \rangle_p, y), \quad (1)$$

where $0 \leq t < p$, $0 \leq m \leq p$, and the notation $\langle j \rangle_p$ means taking the modulus (or integer remainder) of j with respect to division by p . The sum $R(t, 0)$ is taken along column t of the image, whilst the sum $R(t, p)$ is taken along the t^{th} image row. A ghost at N angles means $R(t, m) = 0$ for all t at N values of m .

The discrete projection angle, m , is defined as the gradient of a line, or projection ray, linking pixels located at points m steps across and one pixel down from each other. The projection angle $\tan^{-1}(1/m)$ can be re-mapped to conventional tomographic angles, with one unique angle for each value of m , as the direction of the ray that links nearest neighbour pixels that lie on a ray with the same value of t [10]. Then $\theta_j = \tan^{-1}(y_j/x_j)$, where the vector linking nearest neighbour pixels has relative coordinates (x_j, y_j) . A scheme to perform exact discrete rotations of FRT projections was presented in [16]. Affine transformations, in the spatial domain or in FRT space, can be applied to perform discrete rotations on ghost images. The property of projections summing to zero for N projection angles is retained under these transformations, provided that the affine mapping is invertible.

3 Generating Ghosts by Iteration

Figure 1 shows the principle of iterative ghost generation. An initial ghost, usually corresponding to zero-sum projections along the row or column direction, is chosen. This ghost has a trivial structure, requiring only 2 pixels, one with value $+1$ and the other with value -1 . The separation of these pixels is arbitrary, but to maximise ghost compactness, the two pixels are kept adjacent (Figure 1(a)). This pair of pixels is propagated into other discrete projection directions (for example along the column direction, Figure 1(b), and the diagonal direction, Figure 1(c)). A sign-reversed copy of the initial ghost is added at

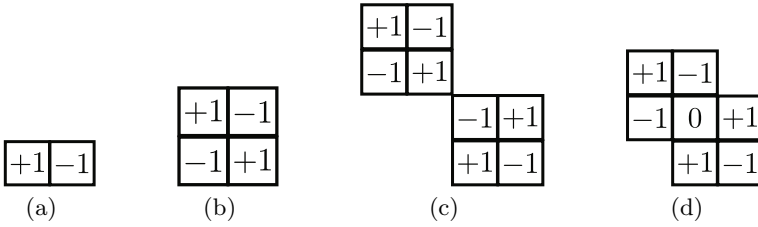


Fig. 1. Pairs of +1 and -1 pixels can be displaced and added iteratively to produce cancellation in new directions. A horizontal ghost a) is shifted vertically b) and then diagonally c) to produce a ghost that is invisible at three projection directions: 0, 45 and 90 degrees. d) The third step can be translated so that one pixel with a +1 value has the same location as a pixel with a -1 value. This reduces the number of image pixels to $2^N - 2$.

some displacement along a new angle, ensuring cancellation of the sums in both original and the new direction is preserved. This process is repeated to include all of the angles for which the ghost must sum to zero. After N steps it is possible to attain a maximum image intensity of 2^N if there is no overlap of the +1 and -1 values at the same pixel location after the dilation.

The sign-reversed ghost can be deliberately translated to ensure the next -1 valued pixel is superimposed on the location of the last +1 valued pixel. This method always cancels at least two of the current ghost pixels and reduces the spatial extent and rate that pixel intensities will increase under subsequent iterations. The rate of growth of pixel intensities in iterated ghost images still remains $O(2^N)$.

4 Ghost Generation by Row vs. Column Oriented Iteration

The FRT, $R(t, m)$, projects data from a symmetric $I(x, y)$ space of size $p \times p$. However the row (x) direction is arbitrarily chosen as the t axis which represents the displacement of each projected ray. The projection angle m accumulates as displacements along the column (y) direction at step $\langle my \rangle_p$. The FRT projections thus turn out to be sensitive to the row or column symmetry of any process we apply.

A simple algorithm is given in Algorithm 1 to describe iterative ghost construction, in an initially blank $p \times p$ 2D image space, $A(x, y)$.

Here the function CS denotes a circular shift of the matrix A . The operation CS moves the origin of the matrix A to the point (x, y) . The pixel locations (and the pixel content) wrap modulus p under circular shifts. The algorithm is iterated $N - 1$ times to produce the set of N required discrete projection angles. Gradient $x_i : y_i$ corresponds to an angle at x_i steps across and y_i steps down.

For row displacements, the circular shift parameters take the values $1 \leq x_i \leq N$ and $y_i = 1$. An example of the ghosts produced by this method is given in

```

begin Ghost Construction
  |  $A \leftarrow 0$  ; //  $A$  is an empty  $p \times p$  array
  | //  $A$  contains initial ghost, here along direction  $m = 0$ 
  |  $A(0,0) = +1$ ;
  |  $A(0,1) = -1$ ;
  | for  $i \leftarrow 1$  to  $N - 1$  do
  | | // Iteratively subtract shifted versions of  $A$  from  $A$ 
  | |  $A = A - CS(A, [x_i : y_i])$ ;
  | end
end

```

Algorithm 1. A simple algorithm describing iterative ghost construction

Figure 2(a) and the FRT of that ghost image is shown in Figure 2(b). Because the displacements are here m steps across for each step down ($m : 1$ gradients), the directions of the zero-sum ghosts correspond to sequential values of m . The sum of the absolute values of all pixels in row-dilated images is always exactly $2N$ after N row-oriented dilations. The spatial extent of the ghost after 8 dilations, as expected, covers 9 full image rows.

For column displacements, the circular shift parameters are $x_i = 1$ and $1 \leq y_i \leq N$. An example of the ghosts produced by this method is given in Figure 3(a) and the FRT of that ghost image is shown in Figure 3(b). Because the shifts here are one step across for m steps down ($1 : m$ gradients), the directions of the zero-sum ghosts are sequential values of $1/m = m^{-1} \pmod{p}$. The angles are hence sequential in the inverse values of m , the integer angle parameter.

The sum of the absolute values of all pixels in column-displaced images after N dilations varies slowly with N , as shown in Table 1. The spatial extent of the ghost after 8 dilations covers 8 image rows (the initial ghost, $m = 0$, is a column entry).

Table 1. Growth of the dynamic range of the absolute values of the ghost pixels after N iterations along the row and column directions. The “sum absolute” figure is the integrated intensity of the absolute value of all the ghost image pixels for each N . The values here are computed for an array size $p = 127$.

Number of ghosts, N	7	8	10	12	16	20
Row Iteration						
min/max	-5/5	-6/8	-20/18	-49/58	-468/526	-4968/5448
Sum absolute	2^7	2^8	2^{10}	2^{12}	2^{16}	2^{20}
Column Iteration						
min/max	-1/1	-1/1	-2/2	-4/4	-21/21	-146/146
Sum absolute	26	40	104	304	3040	34224

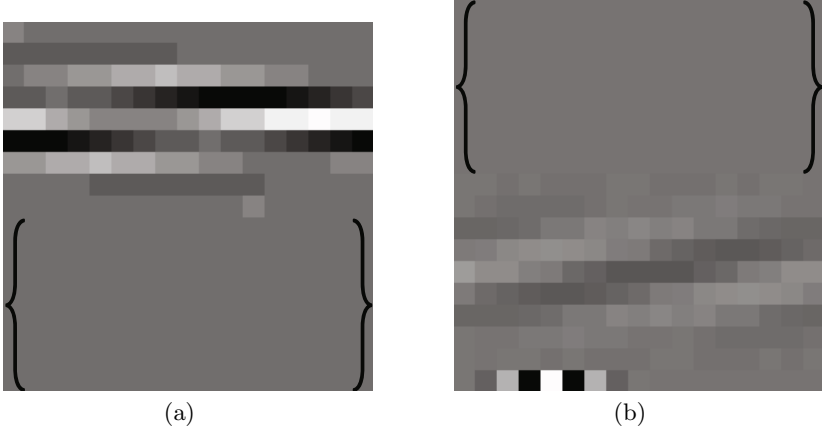


Fig. 2. (a) A 17×17 ghost image $I(x,y)$ with $N = 8$ zero-sum projections created by row-oriented iterated convolutions. (b) the FRT, $R(t,m)$, of (a). The first 8 rows corresponding to $m = 0$ to 7 are zero-sum projections. The pixel values in (a) range from -6 (shown as black) to $+8$ (shown as white). Mid-grey corresponds to a value of 0. The summed pixel values in (b) range from -56 to $+70$. The bracketed regions enclose empty image and projection space, respectively. The large values on the bottom row of (b) are the row sums ($m = p$) of the pixel values shown in the image (a). These values accumulate rapidly for row-oriented displacements.

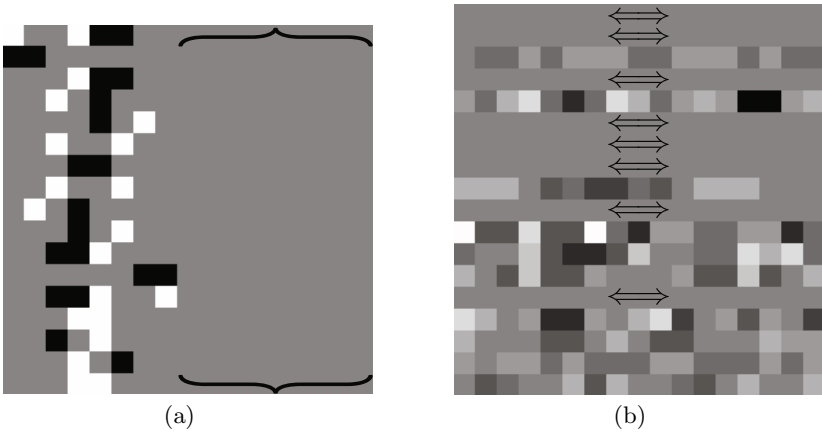


Fig. 3. (a) A 17×17 ghost image $I(x,y)$ with $N = 8$ zero-sum projections created by column-oriented iterated convolutions. (b) The FRT, $R(t,m)$ of (a). The 8 rows corresponding to $m = \{0, 1, 3, 5, 6, 7, 9, 13\}$ are zero-sum projections. The black arrows point to these zero-sum projections. The pixel values in (a) range from -1 (shown as black) to $+1$ (shown as white). Mid-grey corresponds to a value of 0. The summed pixel values in (b) range between -6 to $+8$. The bracketed portion of the image space in (a) is empty.

5 Growth of Iterated Ghosts

Table 1 shows examples of the growth of ghost image pixel values under the two methods outlined above.

The values given in Table 1 are sensitive to p when $N \approx p$ because of the strong overlap of pixel positions after each wrapping. For $p \gg N$, the ghost pattern wraps, at modulus p , around the array in a clean spiral pattern that causes minimal pixel position overlaps. In Table 1, $p = 127$ and the maximum value of N is 20.

A log plot of the mean absolute ghost image pixel value (g_{max}) as a function of the number of ghost angles, N , is well fitted by a straight line. For the column displacements, $\log_2(g_{max}) \approx 0.7(N-9)$. For the row displacements, $\log_2(g_{max}) \approx 0.8(N-5)$. The value of N at which we observe the onset of pixel intensities rise above +1 and -1 is smaller for the row-based dilations. The rate of growth of ghost image pixel values is significantly greater for the row-based approach rather than for column-based iteration.

Examples of ghost images at a larger N and image scale are given in Figure 4. Despite the large range of pixel values required to construct these ghosts, their spatial extent is quite small (being confined to 20 rows or columns respectively for $N = 20$). The small spatial footprint of iterated ghost images may be a significant advantage in many applications [24].

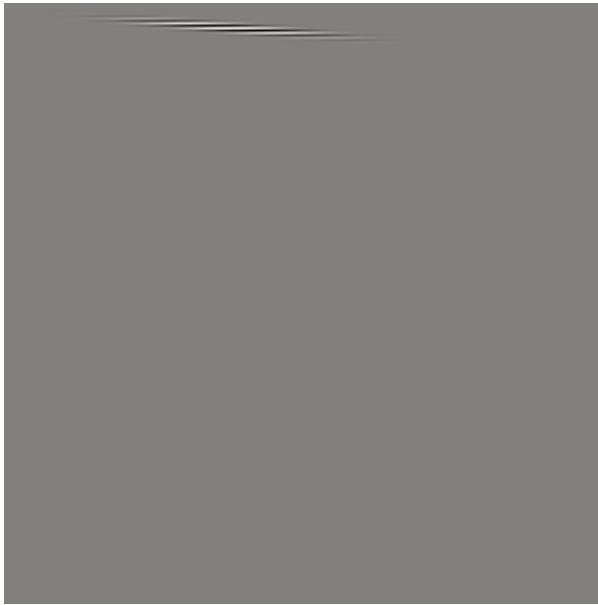
Applying affine mappings (computed modulus p) to N zero-sum projection ghost images such as those in Figure 5 spreads the ghost pixels entries across the image space. However the resultant affine-mapped ghost images still have N zero-sum projections, albeit at a new set of FRT angles.

6 Further Work and Conclusions

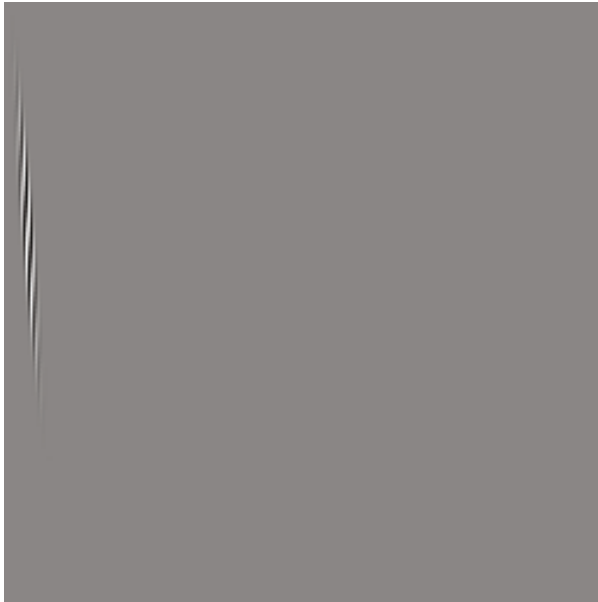
The multi-valued ghosts shown here have been used in “de-ghosting” applications where rows of missing image data can be recovered exactly [42]. There we exploited the fact that the correct data is known to have zero or constant value across redundant parts of the image space. Missing data causes non-zero reconstructions in those regions that should be zero. Missing projection data can be regarded as being comprised of linear combinations of scaled and shifted ghost images.

The ghost images in Figure 4 are reminiscent of the Gabor oriented-filters used in texture analysis. It would be interesting to pursue this connection, as the 1D Fourier transform of each FRT row corresponds directly to a 1D slice in 2D FFT space.

The great advantage of ghosts constructed from only +1 and -1 pixel values is that they can be used to create a mixture of image and anti-image data where the combined image then has zero-sum projections at N angles. This type of “expanded” ghost was used in [13] to encode redundancy into data for forward error correction in a proposed data transmission scheme. For small N ($N < 9$),

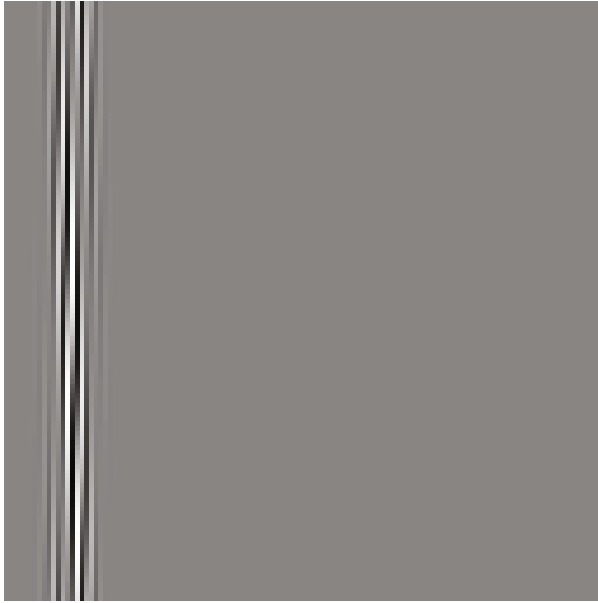


(a)

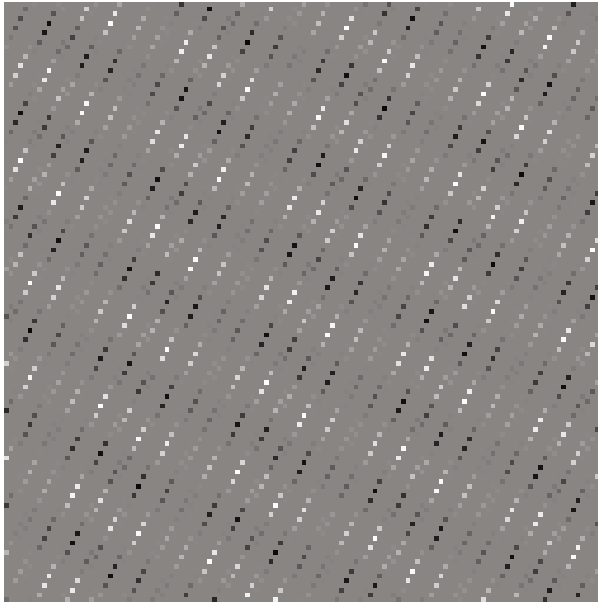


(b)

Fig. 4. Two 251×251 ghost images, each image has $N = 20$ zero-sum projection angles. The row-iterated ghosts in (a) have pixel values that range from -4968 to $+5448$. The values in (b), for column-iterated ghosts, range from -146 to $+146$. The original image intensities have been both normalised to the range $0 - 255$ for display.



(a)



(b)

Fig. 5. A 127×127 ghost image with $N = 29$ zero-sum projections. Here the column-iterated ghosts have wrapped around the array from bottom to top. (b) Affine transform of (a) is still an $N = 29$ ghost image, but at a new set of FRT angles. The same pixel intensities are now mapped uniformly across the image space in (b).

the dilation method yields ghosts with pixel values restricted to ± 1 . For these cases it is possible to keep a single $+1$ pixel at $(0,0)$ and reorder the pixel placements on subsequent rows whilst preserving the ghost properties.

Adding a copy of a negated ghost to a $+1$ pixel position cancels that pixel value to zero. Applying this process repetitively enables creation of a large blank space within the ghost structure that makes it suitable for image/anti-image generation (see Figure 6). The increase in pixel values of the overlaid ghosts causes larger intensities in the anti-image as it is synthesized from scaled image values multiplied by the brightness of the lower components of the ghost data. The addition of displaced ghost images causes further lateral spread of the ghost pixels. This method of creating “clear space” in a ghost image is then limited to clearing rows until the ghost image/anti-image wraps around to the top of the array.

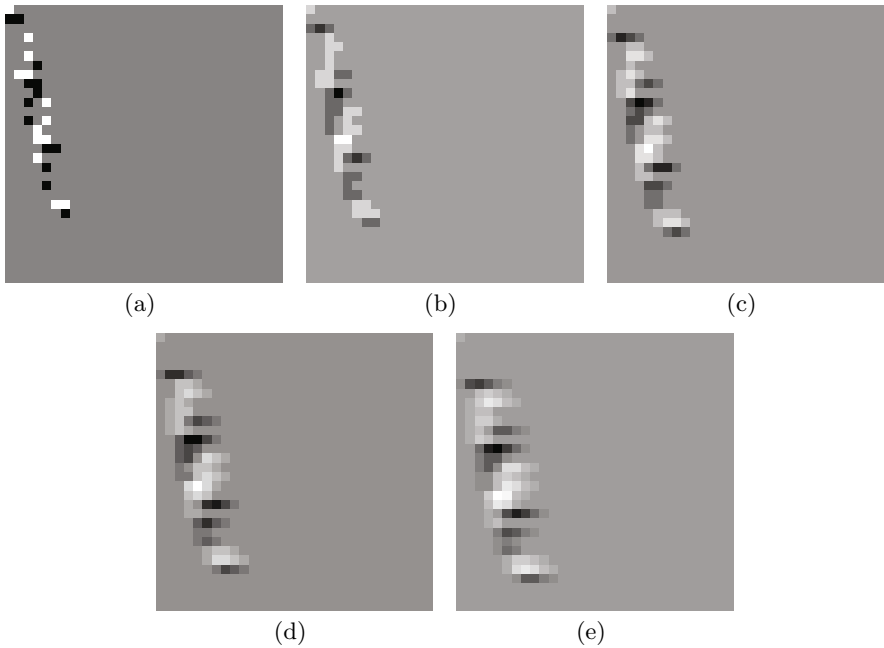


Fig. 6. The $+1$ pixel in the top left corner of the top row of the ghost image is retained in the sequence (a) through (e). Translated copies of image (a) are added to the second row of (a) to produce image (b), which has zero pixel values on row 2. The process is repeated to zero the pixel entries on row 3 in (c), row 4 in (d) and row 5 in (e). The ghost image in (e), being composed of translated and scaled copies of the ghost image in (a) has exactly the same zero-sum projection rows after applying the FRT. The original image intensities in (a) through (e) have been normalised to the range $0 - 255$ for display.

Acknowledgement. S.C. acknowledges the past support of a scholarship funded through the School of Physics and Faculty of Science at Monash University whilst completing his PhD. I.S. acknowledges ongoing support from the School of Physics at Monash for this work.

References

1. Barcucci, E., Del Lungo, A., Nivat, M., Pinzani, R.: X-rays characterising some classes of discrete sets. *Linear Algebra and its Applications* 339, 3–21 (2001)
2. Chandra, S.S., Normand, N., Kingston, A., Guédon, J.P., Svalbe, I.: Fast Mojette Transform for Discrete Tomography. Elsevier Signal Processing Submitted June (in Review), Available on arXiv.org (2010), <http://arxiv.org/abs/1006.1965v1>
3. Chandra, S.S., Svalbe, I.: A Fast Number Theoretic Finite Radon Transform. In: Proceedings of the Digital Image Computing Techniques and Applications Melbourne (December 2009), <http://dx.doi.org/10.1109/DICTA.2009.67>
4. Chandra, S., Svalbe, I.D., Guédon, J.-P.: An exact, non-iterative mojette inversion technique utilising ghosts. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) DGCI 2008. LNCS, vol. 4992, pp. 401–412. Springer, Heidelberg (2008)
5. Gardner, R.J., Gritzman, P.: Discrete tomography: determination of finite sets by x-rays. *Trans. Amer. Math. Soc.* 349(6), 2271–2295 (1997)
6. Grigoryan, A.M.: New algorithms for calculating the discrete Fourier transforms. *J. Vichislit. Matem. i Mat. Fiziki* 25(9), 1407–1412 (1986)
7. Guédon, J.P., Normand, N., Kingston, A., Parrein, B., Servièeres, M., Évenou, P., Svalbe, I., Autrusseau, F., Hamon, T., Bizais, Y., Coeurjolly, D., Boulos, F., Grail, E.: *The Mojette Transform: Theory and Applications*. ISTE-Wiley, Chichester (2009)
8. Herman, G.T., Davidi, R.: Image reconstruction from a small number of projections. *Inverse Problems* 24, 17 (2008)
9. Katz, M.: Questions of Uniqueness and Resolution in Reconstruction from Projections. *Lecture Notes in Biomathematics*. Springer, Heidelberg (1977)
10. Kingston, A., Svalbe, I.: Projective transforms on periodic discrete image arrays. *Advances in Imaging and Electron Physics* 139, 75–177 (2006)
11. Louis, A.K.: Picture reconstruction from projections in restricted range. *Mathematical Methods in the Applied Sciences* 2, 209–220 (1980)
12. Matúš, F., Flusser, J.: Image Representation via a Finite Radon Transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15(10), 996–1006 (1993), <http://dx.doi.org/10.1109/34.254058>
13. Normand, N., Svalbe, I.D., Parrein, B., Kingston, A.M.: Erasure coding with the finite Radon transform. In: IEEE Wireless Communications & Networking Conference, Sydney (April 2010), <http://dx.doi.org/10.1109/WCNC.2010.5506385>
14. Svalbe, I., Normand, N.: Properties of minimal ghosts. In: Accepted for presentation at DGCI Nancy, France, April 2011 (2010)
15. Svalbe, I., Normand, N., Nazareth, N., Chandra, S.: On constructing minimal ghosts. In: APRS Conference, DICTA 2010, pp. 1–3 (December 2010)
16. Svalbe, I.: Exact, scaled image rotation using the Finite Radon Transform. *Pattern Recognition Letters* (2010) (in press), <http://dx.doi.org/10.1016/j.patrec.2010.06.015>
17. Zopf, S.: Construction of switching components. In: Kuba, A., Nyul, L.G., Palagyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 157–168. Springer, Heidelberg (2006)

Properties of Minimal Ghosts

Imants Svalbe¹ and Nicolas Normand^{1,2}

¹ School of Physics, Monash University, Melbourne, Australia

² IRCCyN, University of Nantes, Nantes, France

Abstract. A ghost image is an array of signed pixel values so positioned as to create zero-sums in all discrete projections taken across that image for a pre-defined set of angles. The discrete projection scheme used here is the finite Radon transform. Minimal ghosts employ just $2N$ pixels to generate zero-sum projections at N projection angles. We describe efficient methods to construct N^{th} order minimal ghost images on prime-sized 2D arrays. Ghost images or switching components are important in discrete image reconstruction. Ghosts usually grow larger as they are constrained by more projection angles. When ghosts become too large to be added to an image, image reconstruction from projections becomes unique and exact. Ghosts can be used to synthesize image/anti-image data that will also exhibit zero-sum projections at N pre-defined angles. We examine the remarkable symmetry, cross- and auto-correlation properties of minimal ghosts. The geometric properties of minimal ghost images may make them suitable to embed in data as watermarks.

1 Introduction

Ghosts and switching components play an important role in deciding whether an exact discrete image can be reconstructed from a given set of projections. Switching components enable the data located at certain image positions to be interchanged whilst leaving unchanged one or more projected views of that data. Similarly, ghosts are primitive images that can be added to any existing data set without changing some projected views of that data [11, 14]. A unique image is able to be reconstructed, without prior information or assumptions, when the addition of any ghost to an image is prohibited by the constraints imposed by the projections of that data.

This paper presents an efficient way to construct very large numbers of visually distinct ghost images, each of which has N zero-sum projections. These images, constructed on a $p \times p$ 2D array, where p is prime, are comprised of N pixels with value $+1$ and N pixels with value -1 . Affine transforms can be applied to remould the spatial distribution of the $2N$ pixel entries as desired.

Our first application of these minimal ghosts is to encode redundancy into image data. Ghosts form a direct, causal link between data in image space and projected views of that data in some projection space. Here we purposefully couple original image data with anti-image data built from ghost images. Combining image and anti-image data into the same space guarantees it too will have zero-sum projections at the same pre-selected angles. The location of the anti-image

data is restricted to fall within redundant portions of the original image space (any region where the image has constant or zero values). Redundancy that was designed in projection space is then able to be explicitly encoded into the spatial domain data. This approach has relevance to forward error-correction schemes in redundant data transmission [18].

Our second application of minimal ghosts exploits the strong geometric constraints of locating $2N$ pixels so that they sum to zero along N projection directions. Minimal ghost images have strong symmetry, compact auto-correlation and diffuse cross-correlation properties. Minimal ghost images could serve as low-visibility, high-security and robust watermarks.

Section 2 reviews briefly the discrete projection scheme in which ghost image pixel values sum to zero. Section 3 compares the complex ghost structures created by straight-forward iterated convolutions with the more demanding requirements to synthesize N^{th} order ghosts using $O(N)$ rather than $O(2^N)$ pixels. Observations on the systematics of the minimal ghost generation method are made here. Section 4 gives examples to highlight the strong symmetry properties of minimal ghosts. In Section 5, we use minimal ghosts as the skeletal structure to construct image/anti-image combinations. The combined image/anti-image data inherits the zero-sum projection properties of the ghost image. Section 6 demonstrates the high auto- and low cross-correlations that exist between minimal ghost images of order N . Section 7 summarises our current conclusions and foreshadows extensions to this work.

2 Discrete Projections, Finite Radon and Affine Transforms

The projection scheme used here is that of the finite Radon transform (FRT) [17]. FRT and its inverse are based on the paired transforms developed by Grigoryan [6] that exactly splits a 2D DFT signal into a set of 1D discrete oriented slices. The FRT is a highly symmetrised form of the discrete Mojette projective transform [7].

All information present in an original image $I(x, y)$ is captured, in full, by the set of discrete FRT projections, $R(t, m)$. $I(x, y)$ is hence exactly recoverable from $R(t, m)$. A fast algorithm to compute the FRT projections and to invert them, based on the number-theoretic transform, was presented in [3]. The FRT is well-suited for the systematic encoding of redundant data. This approach was exploited in a proposal for an efficient forward error-correcting internet data-transmission scheme [18]. Ghost images were used for systematic encoding of the data in that work.

The FRT projects data from a $p \times p$ image array, where p is prime, at a set of $p + 1$ discrete projection angles. Each projection is comprised of p parallel translated rays. Each ray sums the intensity of exactly p pixels located across the image domain. The FRT projection rays wrap periodically when they encounter any boundary of the image. The FRT, $R(t, m)$, of an image $I(x, y)$ is defined as:

$$\forall m \in [0, p - 1], R(t, m) = \sum_y I(\langle t + my \rangle_p, y) ; \quad R(t, p) = \sum_x I(x, t). \quad (1)$$

The translate parameter t has $0 \leq t < p$, the projection angle parameter m has $0 \leq m \leq p$. The notation $\langle j \rangle_p$ means taking the modulus (or integer remainder) of j with respect to division by p .

The discrete projection angle, m , is defined as the gradient of a projection ray that links pixels located m steps across and one pixel down from each other. The projection angle $\tan^{-1}(1/m)$ can be re-mapped to conventional tomographic angles, with one unique angle for each value of m [12]. There the nearest-neighbour pixels that belong the same translate, t define the ray direction. The sum $R(t, 0)$ is taken along column t of the image, whilst the sum $R(t, p)$ is taken along the t^{th} image row. A ghost image with N zero-sum projection angles has $R(t, m) = 0$ for all t at N values of m .

A scheme to perform exact, discrete rotations of image data from within the FRT domain was presented in [20]. The property of parallelism is preserved by affine mappings, whether the mapping is applied in image or projection space. Arnold’s cat [1] provided an early example of information-conserving, discrete image mappings. The parallel, zero-sum projected rays in any ghost image will be retained as parallel, zero-sum projections after any invertible affine transformation is applied to that image.

3 Constructing Ghosts and Minimal Ghosts

3.1 An Overview of Ghosts

Non-minimal ghosts can be constructed in a straightforward way using a process of iterated convolutions as has been shown by the earlier work of [2,5,10,23]. Fig. 1a shows a pair of +1 and -1 points that will form a zero-sum projection when added to any row of an image. This primitive shape can be propagated at some new angle, for example along the column directions, as shown in fig. 1b. Adding a displaced copy of the original ghost, with signs reversed, means that the sums taken in the original direction and along the new direction are both guaranteed to be zero, as shown again in fig. 1c, where the cancellation now includes the diagonal direction. This simple propagating process results in a ghost image that contains up to 2^N pixels after displacing the accumulating shapes in N directions. The complexity of iteratively-built ghost structures rises exponentially with N .

The displacements can be overlapped, as shown in fig. 1d, to reduce the number of pixels required to form a ghost image. In a periodic array, a large displacement that makes the image wrap, modulus p , back onto itself can also be used to effect the partial cancellation. A related paper [21] reviews the use of N iterated convolutions to produce ghosts with N zero-sum projection angles. That paper examines the rate of growth of the dynamic range and the increase of spatial extent of ghost images, as a function of N , the number of zero-sum projections.

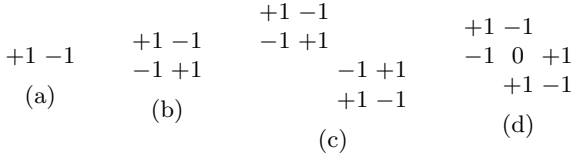


Fig. 1. Groups of $+1$ and -1 pixels displaced in some new direction maintain their cancelling sums in that direction. A horizontal ghost a) is displaced vertically b) and then their union is displaced diagonally c) to produce a ghost that is invisible at three projection directions: 0 , 45 and 90 degrees. d) The third dilation can be displaced so that one pixel with a $+1$ value has the same location as a pixel with a -1 value. This reduces the number of ghost image pixels to $2^N - 2$.

3.2 Minimal Ghost Construction

Our objective here is to produce ghosts with a large number N of zero-sum projections using the smallest possible number of non-zero pixels. Fewer pixels are required if a $+1$ pixel can form a zero-sum with -1 pixels that lie in more than one direction. If each of the N pixels with value $+1$ can be positioned to line up with N pixels of value -1 along exactly the same N directions, then a total of just $2N$ pixels are required to construct an N^{th} order ghost.

An efficient method to generate minimal ghosts was presented in [22]. It relies on finding integer values a , b , c and d for a matrix $T = [a \ b; c \ d]$, such that the matrix T^{2N} , evaluated modulus p , yields the identity matrix $I = [1 \ 0; 0 \ 1]$. This procedure is akin to finding the cyclic primitive roots of a complex vector, quaternion or pseudo-tensor. Such matrices will exist for some values of p , because integers with exponent N are cyclic when they are computed modulus p .

The initial matrix T is used to map the (x_0, y_0) coordinates of (an arbitrarily selected) $+1$ ghost pixel location (here called C_0) to produce the first -1 ghost pixel location (here called S_0). The matrix T^i then remaps that current (x, y) point to the location of the next $+1$ (for i even) or -1 (for i odd) valued ghost pixel location (here called C_i and S_i) respectively. The matrix T^{2N-1} maps the coordinates of the last S_i point back to C_0 , the coordinate of the initial starting point. Examples of T and corresponding C_i and S_i points for some values of N are given in appendix A.1. To check this procedure works, the FRT projection, $R(t, m)$, of the $p \times p$ ghost image is computed and the value of R must be zero, for all t , for N values of m .

At present, we employ a simple brute-force search method to find a candidate matrix T . Not all matrices T for which $T^{2N} = I \pmod{p}$ will produce minimal ghosts. We have not conducted a detailed investigation of the properties of the working matrix solutions, but have observed some of the systematic traits in the solutions for T that do work. For example, T has solutions when $p = 2\alpha N \pm 1$, where α is a positive integer. The largest value of N that leads to non-trivial ghost results is $2N = p + 1$. Solutions for $N = p$ or $p - 1$ can be found, but these lead to the so-called “universal ghosts”, strings of pixels that form 4- or 8-connected lines in image space. Universal ghosts form zero-sums for all but one or two of the $p + 1$ FRT directions [22].

The structure of ghosts becomes simpler in image or projection space as N approaches either 1 or p . This result is strikingly similar to that for the decomposition of open, median and close filters from mathematical morphology into sets of primitive erosions (or dilations) [15,16,19]. In this respect, minimal ghosts with the “median” value of $N = (p \pm 1)/2$ will have the greatest structural complexity. These median-minimal ghosts offer the optimal compromise of having a large number of zero-sum angles, yet maintaining a sparse distribution of ± 1 pixel values spread across the image.

3.3 Systematics of Minimal Ghost Construction

Valid solutions for T are given by $T = [0 \ 1; p - 1 \ x_i]$, where x_i is an element of a set X of integers. Each $x_i \in X$ gives a different, visually distinct, working ghost image solution. Variations of the above form include $T = [n \ 1; nx_i - 1 \ x_i]$ and $T = [0 \ n; n^{-1} \ x_i]$, where n^{-1} is the inverse of n , modulus p . Here the common link is that $\det(T) = 1$. The sum and product, modulus p , of the full set of integers $x_i \in X$ also have interesting properties. For example, for $N = 2p'$, where p' is prime, $\prod x_i = \pm p'$ with the sign fixed by $p' = 4\alpha \pm 1$ for some integer α . When N is even, X contains pairs of complementary values, x_i and $p - x_i$.

The ghost images generated by each solution x_i appear visually distinct, but they are related to each other through affine spatial transforms. The integers x_i depend on N and the array size p for which $T^{2N} = I$. Some examples of the set X that correspond to various N are given in appendix A.2. However the size of the set X , n_X , depends only on the value of N . If N is prime, $N = 2n_X(N) + 1$; if N is a power of a prime, for example p^α , then $n_X = p^{\alpha-1}n_X(p)$. For composite N , for example, $N = i \times j \times k \dots$, then $n_X(N) = (2m - 1)n_X(i) \times n_X(j) \times n_X(k) \dots$, where m is the number of odd factors of N .

Choosing different initial coordinates C_0 , also gives rise to visually different ghost images. Often the N values of m that form zero-sums will fall into distinct, non-overlapping selections of the $p + 1$ possible discrete projection angles. The choice of the initial location (x_0, y_0) for C_0 is arbitrary, provided only that this choice does not lead to degenerate mappings, where the coordinates for $C_{i+1} = K \times C_i$ and $S_{i+1} = K' \times S_i$, for some constants K and K' . Degeneracy results when the same projection angle m satisfies the zero-sum constraint N times.

By symmetry, if the choice $C_0 = (j, k)$ leads to a degeneracy, then so too does $C_0 = (k, j)$. Increased levels of degeneracy occur for the smaller integers in the set X . If (i, j) and (i, k) are degenerate choices for C_0 , we find that $\langle j + k \rangle_p = x_i$. Degenerate mappings do not occur for all N . For example, when $N = (p + 1)/2$, exactly two sets of mutually exclusive zero-sum projection angle sets can form. There are no ‘spare’ projection angles to act as degenerate m ’s.

The size of the set X , its constituency and the degeneracy under the choice of array size p and the C_0 location are all interesting parameters to study from a number-theoretic perspective. These parameters rely on the spacing of prime integers and cyclic permutations of prime sets.

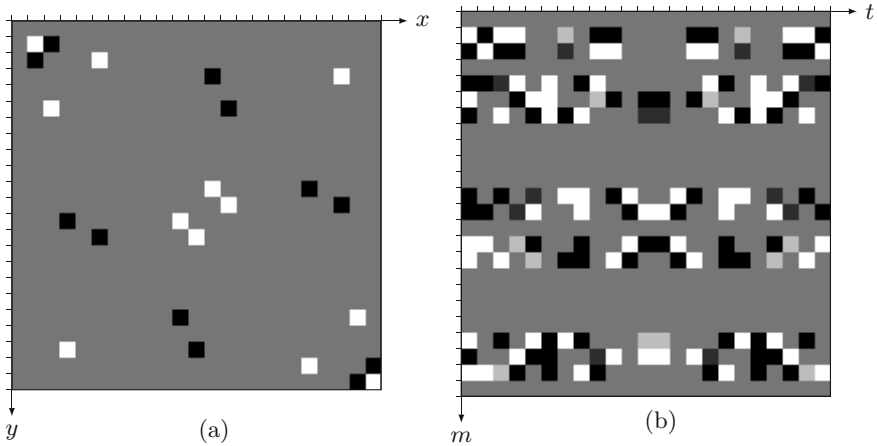


Fig. 2. a) A 23×23 image space $I(x, y)$, containing twelve $+1$ pixels (white) and twelve -1 pixels (black). The pixel locations are designed to make the projected sums of this image vanish at 12 pre-selected angles. b) The 23×24 discrete projections, $R(t, m)$, of the image in a). The projection translate variable, t , is plotted horizontally, with the discrete projection angle, m , shown as the vertical axis. $R(0, 0)$ is at the top left corner of b). Note all the projection values at 12 distinct angles ($m = 0, 3, 7, 8, 9, 10, 13, 16, 17, 18, 19, 23$) have the value zero (shown here as grey).

An example 12^{th} order minimal ghost image is shown in fig. 2a. This 23×23 image contains 12 pixels with value $+1$ (shown as white) and 12 pixels with value -1 (shown as black). Projecting this image results in zero-sums at 12 projection angles, as shown in fig. 2b.

4 Properties of Minimal Ghost Images

The zero-sum properties of a ghost image remain invariant under intensity scaling (for example, using pixel values of $\pm g$ instead of ± 1), under image translation (where addresses wrap modulus p) and affine transforms where the mappings are done modulus p (provided the affine mapping, as expressed in homogenous coordinates, has a proper inverse). The sum of scaled and translated N^{th} order ghost images corresponding to a given set of N projection angles is then also a ghost image at those same N angles.

A most remarkable property of minimal ghosts is their strong symmetry. When N is even, ghost images generate $N/2$ lines that link pairs of the $+1$ points and $N/2$ lines that link pairs of the -1 points. Incredibly, these N lines have a common intersection point that exactly bisects each of these N lines (see fig. 3). For odd N ghosts, the N lines linking pairs of $+1$ and -1 points have a common intersection point that also bisects each of these N lines. This remarkable symmetry recalls the “ n -gon” behaviour of cyclic integers in the Fermat and Carmichael prime number sets 3.

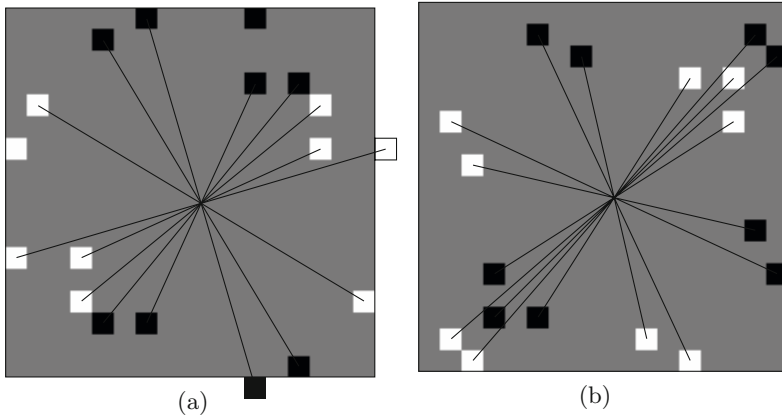


Fig. 3. Minimal ghosts for a) $N = 8$, $x_i = 8$ for $C_0 = (1, 4)$ and b) $N = 9$, $x_i = 3$ for $C_0 = (1, 5)$ on a 17×17 array. Lines joining paired ghost pixel locations are shown. The N lines are all perfectly bisected at a single point of symmetry, located here at $(p/2, p/2)$. The position of two pixels in a) have been wrapped to the opposite edge of the array, modulus p (shown by the box) to complete the symmetry.

For each non-degenerate choice of starting point, C_0 , the symmetry point for a ghost image generated by the T^{2N} method is located at $(p/2, p/2)$. Other symmetry points, for example, $(0, 0)$, $(p/2, 0)$ and $(0, p/2)$ are located between periodic replications of the $p \times p$ ghost image. The N -line bisection symmetry is preserved after applying affine transforms, but can be hard to spot visually. The mean location, (x_N, y_N) , of all of the C_i points is always coincident with the mean location of all of the S_i points. We find this point using $x_N = N^{-1} \sum C_i(x_i)$ and $y_N = N^{-1} \sum C_i(y_i)$ (or, equally, by using the $\sum S_i$ coordinates). Here $N \times N^{-1} \equiv 1 \pmod{p}$.

The ghost images generated by the T^{2N} method always have either odd- or even-reflective symmetry about the leading diagonal. The ghost image of fig. 2a has even symmetry, whilst those of fig. 3 both have odd symmetry. After a transpose operation, the positions of the C_i and S_i pixels are exchanged ($C_i \leftrightarrow S_i$) under odd symmetry or simply swap positions ($C_i \leftrightarrow C_j$, $S_i \leftrightarrow S_j$) under even symmetry.

5 Compacted Ghosts and the Construction of Image/Anti-image Data

Here we warp the structure of a ghost image, like those in fig. 3, using affine transforms. The aim is to isolate a single ghost image pixel as far away as possible from its neighbouring points. This warping opens up a large space in which the targeted point can make maximum use of the ghost translation invariance property. Fig. 4a shows a 127×127 ghost image with $N = 7$. It has a single +1

point at $(0,0)$ and the next non-zero-pixel is located at $(63,33)$. The ghost in fig. 4a can be scaled to some arbitrary intensity and then translated across 127 pixel locations for each of 33 rows. The scaled and translated ghosts can be used to portray an image.

An anti-image is constructed by scaling the N ghost pixels by a different intensity at each translation of the ghost image. The space that follows the clear rows of the first isolated ghost pixel then contains $N - 1$ positive and N negative copies of the image data. The combined image and anti-image data has zero-sums at the same N selected projection angles as the ghost image. Fig. 4b shows an example of an image and anti-image constructed from a 127×33 section, cropped from the Lena image, embedded in a 127×127 space. The ghost here has order $N = 7$. Both images shown in fig. 4 have zero-sum projections at $m = \{5, 38, 60, 68, 85, 102, 110\}$.

The interest here lies in maximising the amount of clear image space that can be created by affine distortion of these minimal ghost images. We have, so far, only applied random affine mappings to the ghost images and then searched to find those ghosts that yield the largest clear row space. The clear-space capacity varies with N and p . For example, at $p = 421$, we can find 130 clear rows when $N = 4$. Table 1 gives values for N_{rows} , the number of empty image rows, as a function of N , the number of zero-sum projections, for $p = 127$.

The solutions given in Table 1 are the result of a non-exhaustive, brute-force search. We intend to apply these compacted ghosts to help symmetrise sets of Mojette projections and hence reconstruct images from asymmetric sets of projection data [4] and for data encoding schemes [18].

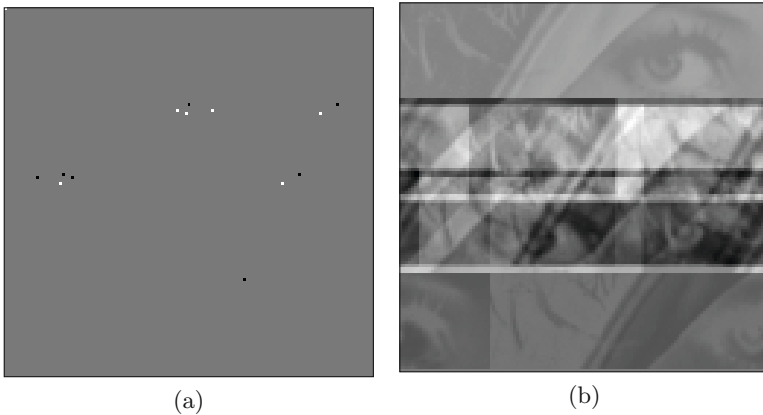


Fig. 4. a) An $N = 7$ compacted ghost in a 127×127 space that exhibits a large clear space between ghost pixels. There are 33 rows of clear image space between the $+1$ pixel at $(0,0)$ and the next -1 ghost image pixel, located at $(63,33)$. b) A 127×127 image comprised of data (rows 0-33) followed by anti-data (rows 34-126). The FRT of the combined image shown in b) is zero at 7 discrete projection angles.

Table 1. N_{rows} , the largest number of completely empty rows, for a 127×127 ghost image space, as a function of N , the number of zero-sum angles

N	3	4	6	7	8	9	14	16	18	21	32	64
N_{rows}	35	33	22	32	30	20	15	14	13	9	7	5

6 2D Correlations between Minimal Ghost Images

There is considerable interest in the generation of 2D sequences that have perfect- or near-perfect correlation properties [8,9,13]. A perfect sequence on an $m \times n$ array has a normalised peak auto-correlation value of $m \times n$ at one location and is zero everywhere else. Near-perfect sequences have off-centre correlation values of ± 1 or ± 2 , rather than zero. The theoretical interest here extends to practical applications in encryption, message-coding and watermarking applications [13].

The remarkable geometric properties of minimal ghosts lead us to examine the correlation properties of ghost images. Ghosts with N zero-sums contain N pixels of $+1$ and N pixels of -1 . The peak value of a $p \times p$ ghost image auto-correlation is then $2N$, considerably less than p^2 .

However these ghost images are sparse and “random” in appearance and so are easy to hide as small perturbations of existing data structures. The linear structure of universal ghosts, where $N \approx p$, makes them less suitable as candidates for watermarks. The largest non-trivial (median) ghost images contain $N = (p + 1)/2$ points. These ghosts provide randomised, high density coverage

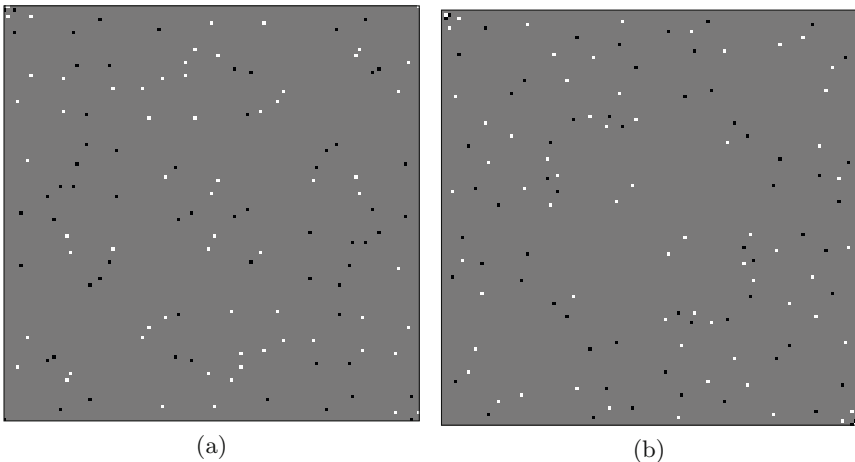


Fig. 5. Ghost images for $N = 64$, $p = 127$ generated by $x_i = 3$ and a) $C_0 = (1, 0)$ and b) for $C_0 = (1, 1)$. The auto-correlation of a) has a peak value of 128 at $(127, 127)$, other locations have absolute values ≤ 2 . The cross-correlation between a) and b) has values $+1, 0$ or -1 .

of the $p \times p$ array space and have a peak auto-correlation value of $p+1$. Although ghost images are generated on $p \times p$ arrays, the ghost images can be regarded as occupying a subset of this space, particularly after applying the compacting affine transforms discussed in section 5.

The strength of the cross-correlation between minimal ghost images depends on whether they have the same value of N , if the ghosts were generated by the same matrix T with the same $x_i \in X$, and if they have the same set of zero-sum angles (i.e. depending on the choice of C_0). The worst case cross-correlations between ghost images give rise to values of ± 4 or less. The cross-correlation of a ghost with a random affine-mapped version of itself is thus restricted to ± 4 or less, independently of the array size p .

The off-peak value of ghost image auto-correlations is ± 2 or less. For example, at $p = 127$ and $N = 64$, the ghost generated by $x_i = 3$ has a peak auto-correlation value of 128, with 3844 pixels of grey value -2, 504 at -1, 55316 at zero, 524 at +1 and 3720 pixels with value +2 (see fig. 5).

7 Conclusions

An N^{th} order ghost image contains pixels with value ± 1 or 0 that are aligned to create zero-sum projections for N discrete projection angles. We have demonstrated the means to generate large numbers of distinct ghost images on 2D prime-sized arrays, with each image containing the minimum possible $2N$ non-zero pixels.

These minimal ghost images exhibit interesting discrete geometric properties. The N lines that link specific pairs of ghost pixels were shown to intersect at a single symmetry point that bisects, exactly, each of these N lines. The shape of ghost images can be warped by affine transforms to maximise or minimize the distance between neighbouring ghost pixels. Compact ghost images permit the construction of significantly-sized blocks of image/anti-image data that inherits zero-sum projection properties from the parent ghost image.

The auto-correlation of an N^{th} order minimal ghost has a peak value of $2N$, with side lobes of ± 2 or less. The peak cross-correlation value for pairs of minimal N^{th} order ghosts can be as low as ± 1 for ghosts that have distinct sets of zero-sum angles, but it is always less than ± 4 . Many ghost images can be nested within the same data region with zero or minimal pixel overlap. The correlation properties of nested and co-located ghost images suggest very flexible and reliable signal recovery strategies. Minimal ghost images may make good data watermarks.

We are keen to further understand the properties of the matrix T and the set X of generating integers that produce ghosts on prime-sized arrays. Using a symbolic mathematics package to explore the elements of the matrices T and T^i has revealed some interesting polynomial patterns. There are strong links between N^{th} order minimal ghost images and N by N Latin squares [22]. Large order Latin squares provide very interesting data encryption prospects.

It would also be useful to develop efficient methods to compact ghost images by purposeful means, rather than, as now, searching through large numbers of

random affine ghost transformations. Finding the maximum and minimum size of the empty space between non-zero ghost pixels is also important. Establishment of a theoretical upper-bound for “collision-free” translation of an N^{th} order ghost in a $p \times p$ image would be an important finding.

The method used here to find minimal ghosts on prime 2D arrays can probably be extended to obtain N^{th} order zero-sums projected across planes or lines in prime 3D cubes and higher dimension arrays. This will also be an interesting topic to pursue.

Acknowledgement. Nicolas Normand worked, from June 2009 to July 2010, in the School of Physics at Monash University, and acknowledges the support of an International Fellowship, granted through the Australian Research Council (LX0989907). Imants Svalbe acknowledges ongoing support for this work from the School of Physics and from the Faculty of Science at Monash University.

References

1. Arnold, V.I., Avez, A.: Problèmes ergodiques de la mécanique classique. Gauthier-Villars, Paris (1967); English translation: Benjamin, New York (1968)
2. Barucci, E., Del Lungo, A., Nivat, M., Pinzani, R.: X-rays characterizing some classes of discrete sets. *Linear Algebra and its Appl.* 339(1-3), 3–21 (2001)
3. Chandra, S., Svalbe, I.D.: A fast number theoretic finite Radon transform. In: DICTA 2009, pp. 361–368. IEEE, Melbourne (2009)
4. Chandra, S., Svalbe, I.D., Guédon, J.-P.: An exact, non-iterative mojette inversion technique utilising ghosts. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) DGCI 2008. LNCS, vol. 4992, pp. 401–412. Springer, Heidelberg (2008)
5. Gardner, R.J., Gritzmann, P.: Discrete tomography: determination of finite sets by X-rays. *Trans. AMS* 349(6), 2271–2295 (1997)
6. Grigoryan, A.M.: New algorithms for calculating discrete Fourier transforms. *Zhurnal Vychislitel’noi Matematiki i Matematicheskoi Fiziki* 26(5), 1407–1412 (1986)
7. Guédon, J.P. (ed.): *The Mojette Transform: Theory and Applications*. Wiley-ISTE, Chichester (2009)
8. Hariharan, R.: Near perfect sequences of odd length. In: *Int. Workshop on Signal Design and its Appl. in Comm.*, pp. 4–7. IEEE, Fukuoka (2009)
9. Hayashi, T.: Zero-correlation zone sequence set construction using an even-perfect sequence and an odd-perfect sequence. *IEICE Trans. on Fundamentals of Electronics, Comm. and Computer Sciences* 90-A(9), 1871–1875 (2007)
10. Herman, G.T., Davidi, R.: Image reconstruction from a small number of projections. *Inverse Problems* 24(4) (2008)
11. Katz, M.B.: Questions of uniqueness and resolution in reconstruction from projections. *Lecture Notes in Biomath.*, vol. 26. Springer, Berlin (1978)
12. Kingston, A.M., Svalbe, I.D.: Projective transforms on periodic discrete image arrays. In: *Adv. in Imaging and Electron Physics*, vol. 139, pp. 75–177. Elsevier, Amsterdam (2006)
13. Kuznetsov, O.: Perfect sequences over the real quaternions. In: *Int. Workshop on Signal Design and its Appl. in Comm.*, pp. 8–11. IEEE, Fukuoka (2009)
14. Louis, A.K.: Picture reconstruction from projections in restricted range. *Mathematical Methods in the Applied Sciences* 2, 209–220 (1980)

15. Maragos, P., Schafer, R.W.: Morphological filters—parts I & II. In: Proceedings of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2002), vol. 35, pp. 1153–1184. IEEE, Los Alamitos (1987)
16. Matheron, G.: Random Sets and Integral Geometry. Wiley, New York (1975)
17. Matúš, F., Flusser, J.: Image representation via a finite Radon transform. IEEE Trans. on Pattern Analysis and Machine Intelligence 15(10), 996–1006 (1993)
18. Normand, N., Svalbe, I.D., Parrein, B., Kingston, A.M.: Erasure coding with the finite Radon transform. In: IEEE Wireless Comm. & Networking Conf., Sydney, pp. 1–6 (April 2010)
19. Serra, J.: Image analysis and mathematical morphology. In: Theoretical Adv., vol. 2. Academic Press, London (1982)
20. Svalbe, I.D.: Exact, scaled image rotations in finite Radon transform space. Pattern Recognition Letters (2010) (in press)
21. Svalbe, I.D., Chandra, S.: Growth of discrete projection ghosts created by iterated dilations. In: DGCI 2011. LNCS, Springer, Nancy (2011)
22. Svalbe, I.D., Normand, N., Nazareth, N., Chandra, S.: On constructing minimal ghosts. In: DICTA 2010, Sydney, Australia (December 2010)
23. Zopf, S.: Construction of switching components. In: Nyström, I., Sanniti di Baja, G., Svensson, S. (eds.) DGCI 2003. LNCS, vol. 2886. Springer, Heidelberg (2003)

A Appendices

A.1

Example lists of C_i and S_i ghost pixel coordinates and their zero-sum gradients, m_i are given below for (left) $p = 17, N = 8, T = [1\ 0; 16\ 8], C_0 = (1, 4)$, see fig. [3a](#) and (right) $p = 17, N = 9, T = [1\ 0; 16\ 3], C_0 = (1, 5)$, see fig. [3b](#).

$C_i = \begin{matrix} 1 & 14 & 0 & 3 & 16 & 3 & 0 & 14 \\ 4 & 6 & 11 & 13 & 13 & 11 & 6 & 4 \end{matrix}$	$C_i = \begin{matrix} 1 & 14 & 12 & 2 & 2 & 12 & 14 & 1 & 10 \\ 5 & 3 & 16 & 7 & 16 & 3 & 5 & 15 & 15 \end{matrix}$
$S_i = \begin{matrix} 4 & 6 & 11 & 13 & 13 & 11 & 6 & 4 \\ 14 & 0 & 3 & 16 & 3 & 0 & 14 & 1 \end{matrix}$	$S_i = \begin{matrix} 5 & 3 & 16 & 7 & 16 & 3 & 5 & 15 & 15 \\ 14 & 12 & 2 & 2 & 12 & 14 & 1 & 10 & 1 \end{matrix}$
$m_i = 1\ 2\ 3\ 5\ 6\ 7\ 9\ 16$	$m_i = 4\ 5\ 7\ 8\ 10\ 12\ 13\ 15\ 16$

A.2

Lists of $x_i \in X$, for some example values of N , at selected values of p are given below.

$N = 7, p = 13: \{3\ 5\ 6\}$	$p = 127: \{61\ 91\ 103\}$
$N = 8, p = 17: \{5\ 8\ 9\ 12\}$	$p = 127: \{42\ 48\ 79\ 85\}$
$N = 9, p = 17: \{3\ 4\ 10\}$	$p = 127: \{53\ 87\ 114\}$
$N = 11, p = 23: \{9\ 12\ 13\ 17\ 19\}$	$p = 131: \{10\ 33\ 53\ 75\ 82\}$
$N = 14, p = 29: \{10\ 12\ 13\ 16\ 17\ 19\}$	$p = 139: \{37\ 46\ 54\ 85\ 93\ 102\}$
$N = 37, p = 73: \{3\ 5\ 8\ 11\ 18\ 20\ 26\ 27\ 29\ 37\ 38\ 40\ 43\ 50\ 51\ 56\ 57\ 66\}$	
$N = 39, p = 79: \{5\ 26\ 32\ 35\ 37\ 41\ 55\ 56\ 58\ 59\ 72\ 76\}$	

Mathematical Morphology on Hypergraphs: Preliminary Definitions and Results

Isabelle Bloch¹ and Alain Bretto^{2,*}

¹ Télécom ParisTech, CNRS LTCI, Paris, France
isabelle.bloch@telecom-paristech.fr

² GREYC CNRS-UMR 6072, Bd Marechal Juin BP 5186, Caen, France
alain.bretto@info.unicaen.fr

Abstract. In this article we introduce mathematical morphology on hypergraphs. We first define lattice structures and then mathematical morphology operators on hypergraphs. We show some relations between these operators and the hypergraph structure, considering in particular duality and similarity aspects.

1 Introduction

Mathematical morphology is a widely used theory for contemporary information processing in various lattice frameworks. Concerning structural information processing, mathematical morphology has been developed on graphs [6,10,15,16], but nothing has been done yet on hypergraphs to the best of our knowledge.

Hypergraphs were introduced in the 60s as a generalization of graphs [1], where edges become hyperedges and can connect more than two vertices, and have then been intensively studied. They have shown their interest in various fields such as computer science, game theory, databases, data mining, optimization [17], image processing and segmentation [4,5].

The aim of this paper is to propose preliminary definitions and results in this context. We consider an hypergraph defining the underlying space: $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ with \mathcal{V} the set of vertices and \mathcal{E} the set of hyperedges. The powerset of \mathcal{V} and \mathcal{E} are denoted by $\mathcal{P}(\mathcal{V})$ and $\mathcal{P}(\mathcal{E})$, respectively. We denote a hypergraph by $H = (V, E)$ with $V \subseteq \mathcal{V}$ and $E \subseteq \mathcal{E}$.

After introducing some more notations and basic concepts related to hypergraphs in Section 2, our first objective (Section 3) is to define a lattice structure (\mathcal{T}, \preceq) on the hypergraphs of \mathcal{H} , with \preceq a partial ordering on \mathcal{T} such that (\mathcal{T}, \preceq) is a complete lattice (suitable structure for mathematical morphology, as shown in [3,7,8,11]). Mathematical morphology operators on hypergraphs are then defined in Section 4. Then, in Section 5, we consider dual hypergraphs and establish some links between morphological dilations and hypergraph duality concepts. Finally, in Section 6 we propose a simple example for computing similarity between hypergraphs based on dilations.

* This work was partially funded by a grant from Institut Télécom / Télécom ParisTech, and was done during the sabbatical stay of A. Bretto at Télécom ParisTech.

2 Basic Concepts on Hypergraphs [1]

A *hypergraph* H denoted by $H = (V, E = (e_i)_{i \in I})$ on a finite set V is a family (which can be a multi-set) $(e_i)_{i \in I}$, (where I is a finite set of indices) of subsets of V called *hyperedges*. Sometimes we will denote V as $V(H)$, and E as $E(H)$. Let $(e_j)_{j \in \{1, 2, \dots, l\}}$ be a sub-family of hyperedges of E . The set of vertices belonging to these hyperedges is denoted by $V(\cup_{j \in \{1, 2, \dots, l\}} e_j)$, and $V(e)$ denotes the set of vertices forming the hyperedge e . When no confusion may arise, we will just denote by e the set of vertices it contains. If $\cup_{i \in I} e_i = V$, the hypergraph is without *isolated vertex* (a vertex x is isolated if $x \in V \setminus \cup_{i \in I} e_i$). By definition the *empty hypergraph* is the hypergraph H_\emptyset such that $V = \emptyset$ and $E = \emptyset$.

Let $H = (V, (e_i)_{i \in I})$ be a hypergraph. A *induced subhypergraph* $H(V')$ of H with $V' \subseteq V$ is a hypergraph defined as $H(V') = (V', (e_i \cap V')_{e_i \cap V' \neq \emptyset})$. The *partial hypergraph* H' of H generated by $J \subseteq I$ is the hypergraph $(V, (e_j)_{j \in J})$. Given a subset $V' \subseteq V$, a *subhypergraph* H' is the partial hypergraph $H' = (V', \{e_i, i \in I \mid e_i \subseteq V'\})$. Without loss of generality we can suppose that the empty hypergraph, $H_\emptyset = (\emptyset, \emptyset)$ is a partial hypergraph, (resp. (induced) subhypergraph) of any hypergraph.

The *star* centered at x is the set of hyperedges containing x , denoted by $H(x)$. The value $d(x) = |H(x)|$ is the *degree* of x .

If the family of hyperedges is a set of subsets of V , we say that H is *without repeated* hyperedge i.e. $i \neq j \iff e_i \neq e_j$. The *rank* of H is the maximum cardinality of a hyperedge. A hypergraph is *linear* if $|e_i \cap e_j| \leq 1$ for $i \neq j$. A *loop* is a hyperedge with a cardinality equal to one. A *simple hypergraph* is a hypergraph $H = (V, E = (e_i)_{i \in I})$ such that: $e_i \subseteq e_j \implies i = j$.

The *dual* H^* of a hypergraph without empty hyperedge and isolated vertex H is a hypergraph whose set of vertices is isomorphic (denoted \simeq) to the set of hyperedges of H , and whose hyperedges are given by X_1, X_2, \dots, X_n where $X_j = \{e_i \mid x_j \in e_i\}$. The transpose A^t of the incidence matrix $A = ((a_{ij}))$ of a hypergraph H (i.e. $a_{ij} = 1$ iff vertex i belongs to hyperedge j) is the incidence matrix of $H^* = (V^* \simeq E, E^* \simeq (H(x))_{x \in V})$: for $v_j^* \in V^*$ and $e_i^* \in E^*$, $v_j^* \in e_i^*$ if and only if $a_{ij} = 1$. Consequently $(H^*)^* = H$. Note that a hypergraph can be equivalently defined as a family (potentially multi-set) of hyperedges on a set of vertices, or as an incidence matrix.

A hypergraph $H = (V, E)$ is *isomorphic* to a hypergraph $H' = (V', E')$ ($H \simeq H'$), if there exist a bijection $f : V \rightarrow V'$ and a permutation π of I such that: $f(V(e_i)) = a_{\pi(i)}$, for $e_i \in E$ and $a_{\pi(i)} \in E'$. The mapping f is then called *isomorphism* of hypergraphs. Note that $H \simeq H'$ if and only if $H^* \simeq H'^*$.

Let $H = (V, E)$ be a hypergraph, $E = (e_1, e_2, \dots, e_m)$. A path P in H from x_{i_1} to $x_{i_{s+1}}$ is an alternated vertex-edge sequence $x_{i_1}, e_{i_1}, x_{i_2}, e_{i_2}, \dots, x_{i_s}, e_{i_s}, x_{i_{s+1}}$ such that $\{x_{i_k}, x_{i_{k+1}}\} \in e_{i_k}$, ($k = i_1, i_2, \dots, i_s$) and $x_{i_k} \neq x_{i_j}$, $e_{i_k} \neq e_{i_j}$ ($i_k \neq i_j$), where s is called the *length* of path P . The distance between vertices x and y , $d(x, y)$ is the minimum length among those of all paths which connect x and y . If for each pair of vertices (x, y) there is a path from x to y , the hypergraph H is said connected.

3 Lattice Structures on Hypergraphs

In this section we define a few lattices on hypergraphs, as the basic algebraic structures on which mathematical morphology operators are then defined.

A lattice on the set of vertices can be simply defined by $\mathcal{T} = (\mathcal{P}(\mathcal{V}), \subseteq)$. This is the classical lattice defined on the powerset of a set, with the standard set inclusion as partial ordering. It is a complete lattice. However is it not really interesting since it does not say anything on the structure of the hypergraph.

A more interesting lattice can be defined, based on closed sets of vertices, involving the stars of vertices. Let $H = (V, E)$ be a hypergraph and let $V' \subseteq V$. We say that V' is a *closed set* if $\forall (x, y) \in V'^2, V(H(x) \cap H(y)) \subseteq V'$. We denote by $\mathcal{C}(H)$ the family of closed sets with the empty set.

Proposition 1. *The structure $(\mathcal{C}(H), \subseteq)$ is a complete lattice. The infimum is $\wedge = \cap$ and the supremum is: $\forall (V', V'') \in \mathcal{C}(H)^2, V' \vee V'' = \cap \{V''' \in \mathcal{C}(H) \mid V' \cup V'' \subseteq V'''\}$, i.e. the intersection of all closed sets containing $V' \cup V''$, and its extension to any family. The smallest element is \emptyset and the largest element is V . Note that $\mathcal{C}(H)$ is a Moore family [3].*

A lattice on the set of hyperedges can be defined by $\mathcal{T} = (\mathcal{P}(\mathcal{E}), \subseteq)$. Again it is a classical complete lattice on the powerset of a set and classical results can be used directly. The next definitions, on the hypergraphs themselves, are probably more interesting.

Lattices on the hypergraphs will allow us to better account for the whole structural information encoded in hypergraphs, considering both vertices and hyperedges in the definition of the lattice structure.

The simplest idea is to consider the inclusion on the powerset of vertices and edges, respectively. Other ideas could be to define a partial ordering based on the notions of induced sub-hypergraph, partial hypergraph and sub-hypergraph.

In all cases, \mathcal{T} is defined as:

$$H = (V, E) \in \mathcal{T} \Leftrightarrow \begin{cases} V \subseteq \mathcal{V} \\ E \subseteq \mathcal{E} \\ \{x \in \mathcal{V} \mid \exists e \in E, x \in e\} \subseteq V \end{cases} \quad (1)$$

The last condition ensures that H is actually a hypergraph, where the hyperedges are sets of vertices of V , and can be equivalently written as $\forall e \in E, V(e) \subseteq V$. There is no equivalent restriction on V if isolated vertices are accepted.

Partial ordering based on the inclusion on the powersets of vertices and hyperedges

Definition 1

$$\forall (H_1, H_2) \in \mathcal{T}^2, H_1 = (V_1, E_1), H_2 = (V_2, E_2), H_1 \preceq H_2 \Leftrightarrow \begin{cases} V_1 \subseteq V_2 \\ E_2 \subseteq E_1 \end{cases} \quad (2)$$

This definition is similar to the one used in [6] for graphs.

Proposition 2. *The following properties hold:*

- \preceq defines a partial ordering on \mathcal{T} .
- The infimum is: $H_1 \wedge H_2 = (V_1 \cap V_2, E_1 \cap E_2)$, and for any family (H_i) : $\bigwedge_i H_i = (\bigcap_i V_i, \bigcap_i E_i)$.
- The supremum is: $H_1 \vee H_2 = (V_1 \cup V_2, E_1 \cup E_2)$, and for any family (H_i) : $\bigvee_i H_i = (\bigcup_i V_i, \bigcup_i E_i)$.
- (\mathcal{T}, \preceq) is a complete lattice, which is moreover sup-generated. Its smallest element is $H_\emptyset = (\emptyset, \emptyset)$ and its largest element is $\mathcal{H} = (\mathcal{V}, \mathcal{E})$. We have $\bigvee \emptyset = \bigwedge \mathcal{T} = H_\emptyset$ and $\bigwedge \emptyset = \bigvee \mathcal{T} = \mathcal{H}$.

Note that it is not complemented (in order to have $E \cup E^c = \mathcal{E}$, we would have to consider in E^c all hyperedges that are not in E , including those which have vertices both in V and in V^c , so (V^c, E^c) would not be a hypergraph in \mathcal{T}).

Partial ordering based on the notion of induced sub-hypergraph

Definition 2. *A partial ordering can be defined from the type of inclusion which is implicit in the definition of induced sub-hypergraph, as:*

$$\forall (H_1, H_2) \in \mathcal{T}^2, H_1 \preceq_i H_2 \Leftrightarrow \begin{cases} V_1 \subseteq V_2 \\ E_1 = \{e \cap V_1 \mid e \in E_2\} \end{cases} \tag{3}$$

i.e. H_1 is the sub-hypergraph induced by H_2 for V_1 .

Proposition 3. *The relation \preceq_i is a partial ordering on \mathcal{T} .*

It might be more suitable (to allow for more frequent comparisons between hypergraphs) to propose a less strict version where E_1 is only required to be included in the set of hyperedges of the induced sub-hypergraph:

Definition 3

$$\forall (H_1, H_2) \in \mathcal{T}^2, H_1 \preceq'_i H_2 \Leftrightarrow \begin{cases} V_1 \subseteq V_2 \\ E_1 \subseteq \{e \cap V_1 \mid e \in E_2\} \end{cases} \tag{4}$$

Proposition 4. *The following properties hold:*

- \preceq'_i is a partial ordering on \mathcal{T} .
- $(\mathcal{T}, \preceq'_i)$ is a complete lattice.
- The infimum is: $H_1 \wedge'_i H_2 = (V_1 \cap V_2, \{e_1 \cap V_2, e_2 \cap V_1 \mid e_1 \in E_1, e_2 \in E_2\} \cap \mathcal{E})$, and a straightforward extension to any family (H_i) .
- The supremum is: $H_1 \vee'_i H_2 = (V_1 \cup V_2, E_1 \cup E_2)$, and its extension to any family (H_i) .
- The smallest element is $H_\emptyset = (\emptyset, \emptyset)$ and the largest element is $\mathcal{H} = (\mathcal{V}, \mathcal{E})$.

Another idea involves isomorphisms, as in the following definition.

Definition 4. *Let \mathbf{H} be the set of isomorphism classes of hypergraphs. A partial order on \mathbf{H} can be defined, for all H_1, H_2 in \mathbf{H} as:*

$$H_1 \leq_f H_2 \iff H_1 \text{ is isomorphic (by } f) \text{ to an induced subhypergraph of } H_2 \tag{5}$$

Proposition 5. *The structure (\mathbf{H}, \leq_f) is a complete lattice. The supremum is $\sup\{H_1, H_2\} = H_1 \vee H_2$ (as in Proposition 2 for \leq), and the infimum $\inf\{H_1, H_2\}$ is the maximum common induced subhypergraph (and their extension to any family).*

Partial ordering based on the notion of partial hypergraph

Definition 5

$$\forall(H_1, H_2) \in \mathcal{T}^2, H_1 \leq_p H_2 \Leftrightarrow \begin{cases} V_1 = V_2 \\ E_1 \subseteq E_2 \end{cases} \tag{6}$$

Proposition 6. *\leq_p is a partial ordering on \mathcal{T} , and (\mathcal{T}, \leq_p) is a complete lattice.*

This is simply a restriction of \leq by considering only the hypergraphs with the same set of vertices, so it will not be further considered.

Partial ordering based on the notion of sub-hypergraph

Definition 6

$$\forall(H_1, H_2) \in \mathcal{T}^2, H_1 \leq_s H_2 \Leftrightarrow \begin{cases} V_1 \subseteq V_2 \\ E_1 = \{e \mid e \in E_2 \text{ and } V(e) \subseteq V_1\} \end{cases} \tag{7}$$

Note that the condition on the hyperedges is stronger than $E_1 \subseteq E_2$ and we may have more hyperedges.

Another possibility would be to define \leq'_s by replacing the equality in the condition on the hyperedges by an inclusion (as for \leq'_i).

Proposition 7. *\leq_s and \leq'_s are partial ordering on \mathcal{T} .*

These partial orderings may be interesting when the notions of (induced) sub-hypergraphs are explicitly involved in the application at hand.

In the following, we use \leq for defining in a general way a partial ordering between two hypergraphs.

4 Mathematical Morphology on Hypergraphs

Algebraic Dilation and Erosion. Once we have a complete lattice, the whole algebraic apparatus of mathematical morphology applies.

Let (\mathcal{T}, \leq) and (\mathcal{T}', \leq') be two complete lattices (which can be any of those defined in Section 3, and do not need to be equal). All the following definitions and results are common to the general algebraic framework of mathematical morphology in complete lattices [3,7,8,11,13].

Definition 7. *An operator $\delta : \mathcal{T} \rightarrow \mathcal{T}'$ is a dilation if: $\forall(x_i) \in \mathcal{T}, \delta(\vee_i x_i) = \vee'_i \delta(x_i)$, where \vee denotes the supremum associated with \leq and \vee' the one associated with \leq' . An operator $\varepsilon : \mathcal{T}' \rightarrow \mathcal{T}$ is an erosion if: $\forall(x_i) \in \mathcal{T}', \varepsilon(\wedge'_i x_i) = \wedge_i \varepsilon(x_i)$, where \wedge and \wedge' denote the infimum associated with \leq and \leq' , respectively.*

All classical properties of mathematical morphology then hold [3,7,8], and are therefore not recalled here.

Structuring Element and Morphological Operations. In classical morphology dilations and erosions can be expressed by means of a set, called structuring element, which defines a neighborhood at each point [12], and this idea has been used for graphs as well [16]. The structuring element “centered” at x is $B_x = \delta(\{x\})$. More generally, the structuring element can be interpreted as a binary relation between two elements, thus enabling the extension of this idea to any lattice.

Defining morphological dilations on hypergraphs calls for canonical decompositions of the elements of the considered lattice.

In the case of the lattice $(\mathcal{P}(\mathcal{V}), \subseteq)$, each subset of vertices V can be trivially decomposed as $V = \cup_{x \in V} \{x\}$, and a morphological dilation then writes $\delta_B(V) = \cup_{x \in V} B_x = \cup_{x \in V} \delta(\{x\})$.

In the case of the lattice $(\mathcal{P}(\mathcal{E}), \subseteq)$, each subset of hyperedges E can be decomposed as $E = \cup_{e \in E} \{e\}$, and a morphological dilation is then $\delta_B(E) = \cup_{e \in E} B_e = \cup_{e \in E} \delta(\{e\})$.

Let us now consider the lattice of hypergraphs, with the partial ordering \preceq (see Definition 1). Let $H = (V, E)$ be a hypergraph of this lattice. For E , a natural decomposition consists of $E = \vee_{e \in E} \{e\}$. For V the decomposition should be consistent with the one of E , in order to associate an “elementary” hypergraph to each e . We thus consider $V(e)$, the set of vertices associated with e . Additionally, the decomposition should also involve all vertices that do not belong to any hyperedge. We denote by $V \setminus E$ this set of vertices. Finally we propose the following canonical decomposition of H , from its sup generating property: $H = (\vee_{e \in E} (V(e), \{e\})) \vee (\vee_{x \in V \setminus E} (\{x\}, \emptyset))$.

The question of how the structuring element should be defined depends on the application and on the type of desired results. Examples are provided next.

Example 1. Let us consider $\mathcal{T} = (\mathcal{P}(\mathcal{E}), \subseteq)$. An example of structuring element, defining the elementary dilation of each hyperedge, consists in taking all hyperedges which have at least one vertex in common with the considered hyperedge:

$$\forall e \in E, B_e = \delta(\{e\}) = \{e' \in \mathcal{E} \mid V(e) \cap V(e') \neq \emptyset\}, \tag{8}$$

where the intersection applies on the sets of vertices defining e and e' . Dilating a subset E by this structuring element means adding all hyperedges that are directly connected to E .

As an illustration, let us consider the two hypergraphs depicted in Figure 1. For the first one, we have for instance $\delta(e_1) = \{e_1, e_2, e_3, e_4\}$, $\delta(e_2) = \{e_1, e_2, e_3\}$, and for the second one, $\delta(e_i) = \{e_1, e_2, e_2\}$, for $i = 1, 2, 3$.

Example 2. Another example, where less hyperedges are added, can be obtained by imposing a minimal cardinality on the intersection: $\forall e \in E, B_e^k = \{e' \in \mathcal{E} \mid |V(e) \cap V(e')| \geq k\}$.

Example 3. Let us now consider dilations from $\mathcal{T} = (\mathcal{P}(\mathcal{E}), \subseteq)$ into $\mathcal{T}' = (\mathcal{P}(\mathcal{V}), \subseteq)$. This will be useful later on when considering dual hypergraphs (see Section 5). Then the elementary dilation should map a hyperedge to a subset of

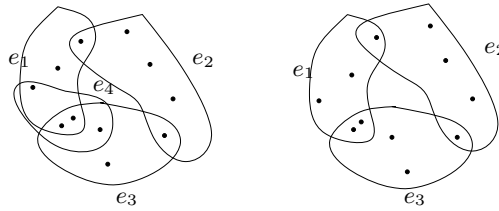


Fig. 1. Two hypergraphs $(V, E_1), (V, E_2)$, defined on the same set of vertices. Hyperedges are displayed as sets of vertices.

vertices. A trivial example is: $\forall e \in E, B_e = \delta(\{e\}) = \{x \in \mathcal{V} \mid x \in e\} = V(e)$, where again e is considered as a subset of vertices. This achieves the required mapping, but does not really dilate anything.

Example 4. More interesting, we can define a structuring element as in Example 1, but considering the resulting subset of vertices: $\forall e \in E, B_e = \delta(\{e\}) = \{x \in \mathcal{V} \mid \exists e' \in \mathcal{E}, x \in e' \text{ and } V(e) \cap V(e') \neq \emptyset\} \cup \{V(e') \mid V(e') \cap V(e) \neq \emptyset\}$. As in Example 2, we could add more strict constraints on the intersection, if we want the dilation to include less vertices.

Example 5. Let us now consider $\mathcal{T} = (\{H = (V, E)\}, \preceq)$. An elementary dilation can be defined according to the proposed canonical decomposition as: $\forall x \in V \setminus E, \delta(\{x\}, \emptyset) = (\{x\}, \emptyset)$, for isolated vertices, and for elementary hypergraphs associated with hyperedges: $\forall e \in E, \delta(V(e), \{e\}) = (\cup\{V(e') \mid V(e') \cap V(e) \neq \emptyset\}, \{e' \in \mathcal{E} \mid V(e') \cap V(e) \neq \emptyset\})$.

Note that if we consider also attributes on the vertices (or hyperedges), other examples can be provided by using a similarity between attributes. For instance isolated vertices could be dilated by adding all vertices that have similar attribute values.

5 Dualities

In the sequel $\delta(\{u\})$ will be simply denoted by $\delta(u)$.

Let $H = (V, E)$ be a hypergraph with $V \neq \emptyset, E \neq \emptyset$, and let $H^* = (V^*, E^*)$ its dual. Let also

$$\delta : V \longrightarrow \mathcal{P}(V)$$

be a mapping. From this mapping we define another one as:

$$\begin{aligned} V &\longrightarrow \mathcal{P}(V) \\ x &\longmapsto \delta^*(x) = \{y \in V; x \in \delta(y)\} \end{aligned}$$

Let us notice that we can also define δ^{**} as

$$\begin{aligned} V &\longrightarrow \mathcal{P}(V) \\ x &\longmapsto \delta^{**}(x) = \{y \in V; x \in \delta^*(y)\} \end{aligned}$$

The following proposition establishes basic results which will be useful next for deriving other results on duality, an important concept on hypergraphs. A particularly interesting result is the one expressed in Corollary [III](#) at the end of this section, linking morphological operators, derived rough spaces, and probability distributions.

Proposition 8. *Let $H = (V, E)$ be a hypergraph with $V \neq \emptyset$, $E \neq \emptyset$ and δ and δ^* as introduced above; we have:*

- a) *for all $X \in \mathcal{P}(V)$, $\delta^*(X) = \bigcup_{x \in X} \delta^*(x) = \{y \in V, X \cap \delta(y) \neq \emptyset\}$ (resp. $\delta(X) = \bigcup_{x \in X} \delta(x) = \{y \in V, X \cap \delta^*(y) \neq \emptyset\}$) iff δ^* is a dilation (resp. δ is a dilation);*
- b) *for all $X \in \mathcal{P}(V)$, if $\bigcup_{x \in X} \delta^*(x) = V$ (resp. $\bigcup_{x \in X} \delta(x) = V$) then $X \subseteq \bigcup_{X \cap \delta^*(y) \neq \emptyset} \delta^*(y)$ (resp. $X \subseteq \bigcup_{X \cap \delta(y) \neq \emptyset} \delta(y)$);*
- c) *$\delta^{**} = \delta$ on V ;*
- d) *if δ^{**} and δ are dilations then $\delta^{**} = \delta$.*

Proof. a) Assume that δ^* is a dilation. The first equality is obvious by definition. Let us show the second one. Let $y \in \delta^*(X) = \bigcup_{x \in X} \delta^*(x)$ then there is a $x \in X$ such that $y \in \delta^*(x) \iff x \in \delta(y)$, so $y \in \{z \in V, X \cap \delta(z) \neq \emptyset\}$.

Let $X \in \mathcal{P}(V)$, and $y \in \{z \in V, X \cap \delta(z) \neq \emptyset\}$, there is $x \in X$ such that $x \in \delta(y) \iff y \in \delta^*(x)$, consequently $y \in \bigcup_{x \in X} \delta^*(x)$.

Conversely, if the equalities hold, then it follows from the first one that δ^* commutes with the supremum, and is hence a dilation.

b) Obvious.

c) Let $z \in \delta^{**}(x)$ then $x \in \delta^*(z)$, and therefore $z \in \delta(x)$.

In the same way $z \in \delta(x) \implies x \in \delta^*(z)$ and $z \in \delta^{**}(x)$. So $\delta^{**} = \delta$ on V .

d) From the definition of a dilation.

Let $\delta : \mathcal{P}(V) \longrightarrow \mathcal{P}(V)$ be a dilation. It gives rise to a hypergraph $H_\delta = (V, (\delta(x))_{x \in V})$, where $\delta(x)$ is seen as a hyperedge built by the vertices defining $\delta(x)$. Now, let $H = (V, E)$ be a hypergraph. We can associate a dilation to H , for instance by considering the following structuring function:

$$\begin{aligned} V &\longrightarrow \mathcal{P}(E) \\ x &\longmapsto \delta(x) = \{e \in E; x \in e\} \end{aligned}$$

Proposition 9. *Let $\delta : V \longrightarrow \mathcal{P}(V)$ be a mapping and $H = (V, E)$ be an hypergraph ($V \neq \emptyset$, $E \neq \emptyset$) without isolated vertex and without repeated hyperedge. We have: $H \simeq H_\delta \iff H^* \simeq H_{\delta^*}$.*

Proof. Suppose that $H \simeq H_\delta$. Because H is without repeated hyperedge, if $x \neq y$ then $\delta(x) \neq \delta(y)$, i. e. $\delta(x) = \delta(y)$ implies that $x = y$, so δ is injective on V .

Let $H = (V; E)$ and $H_\delta = (V_\delta = V; E_\delta = (\delta(x))_{x \in V})$ be hypergraphs. We have: $H \simeq H_\delta \iff$ there a bijection $f : V \longrightarrow V_\delta$ such that $e \in E \iff f(e) = \delta(x) \in E_\delta$, $x \in V$. Since H_δ has no repeated hyperedge, notice that $(\delta(x))_{x \in V}$ is a set $\{\delta(x), x \in V\}$.

It is known that $H \simeq H_\delta \iff H^* \simeq H_\delta^*$, with $H^* = (V^* \simeq E; E^* \simeq (H(x)_{x \in V}))$, $H_\delta^* = (V_\delta^* \simeq (\delta(x))_{x \in V}); E_\delta^* \simeq (H(x)_{x \in V})$. It is sufficient to show that $H_\delta^* \simeq H_{\delta^*}$, with $H_{\delta^*} = (V_{\delta^*} \simeq V; E_{\delta^*} \simeq (\delta^*(x))_{x \in V})$. Let g be a correspondence defined by:

$$\begin{aligned} \{\delta(y), y \in V\} &\longrightarrow V \\ \delta(x) &\longmapsto g(\delta(x)) = x \end{aligned}$$

Since δ is injective, we have $\delta(x) = \delta(y) \implies x = y = g(\delta(x)) = g(\delta(y))$, this correspondence is well defined i. e. it is a mapping.

Clearly g is surjective; moreover g is injective since $|\{\delta(y), y \in V\}| = |V|$. Hence g is a bijection.

Now, $H(x) \in E_\delta^* \iff H(x) = \{\delta(u_i), x \in \delta(u_i)\} = \{\delta(u_1), \delta(u_2), \dots, \delta(u_k)\} \in E_\delta^* \iff g(H(x)) = \{g(\delta(u_i)), i \in \{1, 2, 3, \dots, k\}\} = \{u_1, u_2, \dots, u_k\} = \delta^*(x)$, because $x \in \delta(u_i) \iff u_i \in \delta^*(x)$.

Hence $H(x) \in E_\delta^* \iff g(H(x)) = \delta^*(x) \in E_{\delta^*}$. So $H_\delta^* \simeq H_{\delta^*}$, and finally $H \simeq H_\delta \iff H^* \simeq H_{\delta^*}$.

Proposition 10. *Let $H^* = (V^*, E^*)$ be a hypergraph and let $P = (p_i)_{i \in \{1, 2, \dots, t\}}$ be a discrete probability distribution on V^* , taking rational values. This probability distribution gives rise to a dilation, (resp. a erosion).*

Let δ be a dilation on V^ , then this dilation gives rise to a discrete probability distribution on V^* .*

Proof. To prove the proposition, we will exhibit a particular dilation from P , and respectively a particular probability distribution from a dilation.

Let $P = (p_i)_{i \in \{1, 2, \dots, t\}}$ be a discrete probability distribution with rationale values on V^* . For all $i \in \{1, 2, \dots, t\}$ there are $a_i, b_i \in \mathbb{N}$, $b_i \neq 0$, such that $p_i = \frac{a_i}{b_i} =$

$$\frac{a_i |V^*|}{b_i |V^*|} = \frac{a_i |V^*|}{|V^*|} \cdot \frac{1}{b_i}.$$

We have:

$$\begin{aligned} 1 &= \sum_i p_i = \sum_i \frac{a_i |V^*|}{|V^*|} = \\ &= \sum_i \left(\frac{a_i \cdot |V^*|}{|V^*|} \right) + \frac{|V^*| - \sum_j \left(\frac{a_j \cdot |V^*|}{|V^*|} \right)}{|V^*|} = \sum_i \left(\frac{\lfloor \frac{a_i}{b_i} \cdot |V^*| \rfloor + |V^*| - \sum_j \lfloor \frac{a_j}{b_j} \cdot |V^*| \rfloor}{|V^*|} \right). \end{aligned}$$

$$\text{Let } V_1^* = \{x_1^*, x_2^*, \dots, x_{\lfloor \frac{a_1}{b_1} \cdot |V^*| \rfloor}^*\}, V_2^* = \{x_{\lfloor \frac{a_1}{b_1} \cdot |V^*| \rfloor + 1}^*, \dots, x_{\lfloor \frac{a_1}{b_1} \cdot |V^*| \rfloor + \lfloor \frac{a_2}{b_2} \cdot |V^*| \rfloor}^*\} \dots$$

$V_{t+1}^* = V^* \setminus \bigcup_{i=1}^t V_i^*$. Without loss of generality, we can assume that $V_i^* \neq \emptyset$ for all $i \in \{1, 2, \dots, t+1\}$. By construction we have: $V_i^* \cap V_j^* = \emptyset$ for all $i, j \in \{1, 2, \dots, t+1\}, i \neq j$. Consequently $(V_i^*)_{i \in \{1, 2, \dots, t+1\}}$ is a partition of V^* .

The hypergraph $H^* = (V^*, E^*)$ can be seen as a dual of a hypergraph $H = (V, E)$. Because $V^* \simeq E \iff \bigcup_{i=1}^{t+1} V_i^* \simeq \bigcup_{i=1}^{t+1} \overline{E}_i$, $K = (\overline{E}_i)_{i \in \{1, 2, \dots, t+1\}}$ is a partition of E , where \overline{E}_i is a subset of E , dual of V_i^* . Let us define for $A \subseteq E$

$$\varepsilon(A) = \{\overline{E}_i \in K; \overline{E}_i \subseteq A\} \text{ and } \delta(A) = \{\overline{E}_i \in K, \overline{E}_i \cap A \neq \emptyset\}.$$

It is easy to verify that ε is an erosion and δ a dilation from $(\mathcal{P}(E), \subseteq)$ into $(\mathcal{P}(\mathcal{P}(E)), \subseteq)$.

Now, let δ be a dilation on V^* ; the relation $x_i^* \sim_\delta y_j^* \iff \delta(x_i^*) = \delta(y_j^*)$ is an equivalence relation on V^* . We then denote by $\overline{V^*}_i$ the equivalence classes:

$$V^* / \sim_\delta = \{\overline{V^*}_i; i \in \{1, 2, \dots, t\}\}.$$

Let us now define $p_i = \frac{|\overline{V^*}_i|}{|V^*|}$. We then have $0 \leq p_i \leq 1$ for all $i \in \{1, 2, \dots, t\}$ and $\sum_i p_i = 1$, thereby $(p_i)_{i \in \{1, 2, \dots, t\}}$ is a discrete probability distribution.

This proposition is also interesting to establish links with rough sets. The definition of lower and upper approximations in terms of erosion and dilation, and the equivalence with rough sets have been developed in [2,3]. This result extends these notions to the case of hypergraphs, and $\varepsilon(A)$ and $\delta(A)$ exhibited in the proof are then lower and upper approximations of A in a rough space (this is close to the approach proposed in [14]). Moreover, it adds a link with probabilities.

Corollary 1. *Any discrete distribution of probability on V^* gives rise to a rough space on E .
Conversely any rough space on E gives rise to a discrete distribution of probability on V^* .*

6 Hypergraph Similarity Based on Dilations

As an example of using mathematical morphology on hypergraphs, we briefly propose a notion of similarity between hypergraphs, based on dilations. It is well known that hypergraphs can be used to model several types of networks, such as biological, computer science, semantic networks [9,18,19]. One of the most important tasks is to compare two networks. This comparison can be done using isomorphisms. However, there are two main drawbacks related to the use of isomorphisms:

- the first one concerns tractability, since there is no efficient algorithm to produce an isomorphism between two hypergraphs;
- the second one is that the isomorphism assumption is too rigid, and does not allow considering two hypergraphs as similar if they are not strictly isomorphic.

So we propose to define a new type of “comparator” between hypergraphs, based on dilation, which allows to introduce some “tolerance” for comparing sets of hyperedges, defining a similarity as a degree of overlap between dilated sets of hyperedges.

For any hypergraph (V, E) , we define a dilation on the hyperedges E , for example as:

$$\begin{aligned} \mathcal{P}(E) &\longrightarrow \mathcal{P}(E) \\ A &\longmapsto \delta(A) = \{e \in E; V(A) \cap e \neq \emptyset\} \end{aligned}$$

In the sequel we suppose that if $\delta(A) = \emptyset$ then $A = \emptyset$ (this typically holds when δ is extensive).

Let $H^1 = (V, E^1)$ and $H^2 = (V, E^2)$ be two hypergraphs without empty hyperedge and δ_{E^1} and δ_{E^2} dilations defined on the set of hyperedges of H^1 and H^2 , respectively. We define a *similarity* function s by:

$$\begin{aligned} \mathcal{P}(E^1) \times \mathcal{P}(E^2) \setminus (\emptyset, \emptyset) &\longrightarrow \mathbb{R}^+ \\ (A, B) &\longmapsto s(A, B) = \frac{|\delta_{E^1}(A) \cap \delta_{E^2}(B)|}{|\delta_{E^1}(A) \cup \delta_{E^2}(B)|} \end{aligned}$$

As an illustration, let us consider again the example in Figure 11, with the definition of dilation as in Equation 8. We have quite high similarity values, which fit with the intuition, although the hypergraphs are not isomorphic: $s(e_1, e_i) = \frac{3}{4}$, $i \in \{1, 2, 3\}$; $s(e_2, e_i) = \frac{3}{3}$, $i \in \{1, 2, 3\}$; $s(e_3, e_i) = \frac{3}{4}$, $i \in \{1, 2, 3\}$; $s(e_4, e_i) = \frac{2}{3}$, $i \in \{1, 2, 3\}$; $s(\{e_1, e_2\}; B) = s(\{e_1, e_3\}; B) = s(\{e_1, e_4\}; B) = s(\{e_2, e_3\}; B) = s(\{e_2, e_4\}; B) = s(\{e_3, e_4\}; B) = \frac{3}{4}$, for $B \subseteq E_2$, $B \neq \emptyset$; $s(\{e_1, e_2, e_3\}; B) = s(\{e_2, e_3, e_4\}; B) = s(\{e_1, e_3, e_4\}; B) = s(\{e_1, e_2, e_4\}; B) = \frac{3}{4}$, for $B \subseteq E_2$, $B \neq \emptyset$; and $s(\{e_1, e_2, e_3, e_4\}; B) = \frac{3}{4}$, for $B \subseteq E_2$, $B \neq \emptyset$.

Proposition 11. *Let $H^1 = (V, E^1)$ and $H^2 = (V, E^2)$ be two hypergraphs without empty hyperedge, and δ_{E^1} and δ_{E^2} extensive dilations (i.e. for each hyperedge e , we have $e \in \delta_{E^i}(e)$) defined on E^1 and E^2 . We have the following properties:*

- a) $\forall (e_i, e_j) \in E^1 \times E^2, s((e_i, e_j)) = 0 \iff E^1 \cap E^2 = \emptyset$;
- b) $\forall (e_i, e_j) \in E^1 \times E^2, s((e_i, e_j)) = 1 \implies E^1 = E^2$,
- c) s is symmetrical.

Proof. a) $\forall (e_i, e_j) \in E^1 \times E^2, s((e_i, e_j)) = 0 \iff \forall (e_i, e_j) \in E^1 \times E^2, \delta_{E^1}(e_i) \cap \delta_{E^2}(e_j) = \emptyset \implies \forall (e_i, e_j) \in E^1 \times E^2, e_i \notin \delta_{E^2}(e_j)$ and $e_j \notin \delta_{E^1}(e_i)$ hence $E^1 \cap E^2 = \emptyset$. Indeed, since δ_{E^2} is extensive, $e_j \in \delta_{E^2}(e_j)$ and $\cup_{e_j} \delta_{E^2}(e_j) = E^2$, and therefore having $e_i \notin \delta_{E^2}(e_j)$ for all e_j implies $e_i \notin E^2$. Similarly $e_j \notin E^1$. Conversely, if $E^2 \cap E^1 \neq \emptyset$, then $\mathcal{P}(E^1) \cap \mathcal{P}(E^2) \neq \emptyset$ and any $\delta_{E^1}(A)$ is disjoint from any $\delta_{E^2}(B)$.

b) $\forall (e_i, e_j) \in E^1 \times E^2, s((e_i, e_j)) = 1 \iff \forall (e_i, e_j) \in E^1 \times E^2 \delta_{E^1}(e_i) = \delta_{E^2}(e_j)$. Since δ is extensive, $\forall e_i \in E^1, e_i \in \delta_{E^1}(e_i)$, hence $e_i \in \delta_{E^2}(e_i)$ and therefore $e_i \in E^2$. Similarly $\forall e_j \in E^2, e_j \in E^1$. Therefore $E^1 = E^2$.

c) The symmetry of s is straightforward.

7 Conclusion

In this article we introduced mathematical morphology on hypergraphs. To show the relevance of this relationship between these two domains, we have exhibited a notion of duality in mathematical morphology which corresponds to the concept of duality which is important in the theory of hypergraphs. Other properties of hypergraphs can undoubtedly be expressed using morphological operators (probably such as transversal of a subset of hyperedges of a hypergraph, matching contained in a subset of vertices of a hypergraph). Future work will aim on the one hand at exploring such properties, and on the other hand at studying in more depth the concept of similarity.

References

1. Berge, C.: *Hypergraphs*. Elsevier Science Publisher, The Netherlands (1989)
2. Bloch, I.: On Links between Mathematical Morphology and Rough Sets. *Pattern Recognition* 33(9), 1487–1496 (2000)
3. Bloch, I., Heijmans, H., Ronse, C.: *Mathematical Morphology*. In: Aiello, M., Pratt-Hartman, I., van Benthem, J. (eds.) *Handbook of Spatial Logics*, ch. 13, pp. 857–947. Springer, Heidelberg (2007)
4. Bretto, A., Azema, J., Cherifi, H., Laget, B.: *Combinatorics and image processing. CVGIP: Graphical Model and Image Processing* 59(5), 265–277 (1997)
5. Bretto, A., Cherifi, H., Aboutajdine, D.: *Hypergraph imaging: an overview*. *Pattern Recognition* 35(3), 651–658 (2002)
6. Cousty, J., Najman, L., Serra, J.: *Some morphological operators in graph spaces*. In: Wilkinson, M.H.F., Roerdink, J.B.T.M. (eds.) *ISMM 2009. LNCS*, vol. 5720, pp. 149–160. Springer, Heidelberg (2009)
7. Heijmans, H.J.A.M.: *Morphological Image Operators*. Academic Press, Boston (1994)
8. Heijmans, H.J.A.M., Ronse, C.: *The Algebraic Basis of Mathematical Morphology – Part I: Dilations and Erosions*. *Computer Vision, Graphics and Image Processing* 50, 245–295 (1990)
9. Klamt, S., Haus, U.U., Theis, F.: *Hypergraphs and cellular networks*. *PLoS Comput. Biol.* 5(5) (2009)
10. Meyer, F., Stawiaski, J.: *Morphology on graphs and minimum spanning trees*. In: Wilkinson, M.H.F., Roerdink, J.B.T.M. (eds.) *ISMM 2009. LNCS*, vol. 5720, pp. 161–170. Springer, Heidelberg (2009)
11. Ronse, C., Heijmans, H.J.A.M.: *The Algebraic Basis of Mathematical Morphology – Part II: Openings and Closings*. *Computer Vision, Graphics and Image Processing* 54, 74–97 (1991)
12. Serra, J.: *Image Analysis and Mathematical Morphology*. Academic Press, New-York (1982)
13. Serra, J. (ed.): *Image Analysis and Mathematical Morphology, Part II: Theoretical Advances*. Academic Press, London (1988)
14. Stell, J.: *Relational Granularity for Hypergraphs*. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) *RSCTC 2010. LNCS (LNAI)*, vol. 6086, pp. 267–276. Springer, Heidelberg (2010)
15. Ta, V.-T., Elmoataz, A., Lézoray, O.: *Partial difference equations over graphs: Morphological processing of arbitrary discrete data*. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) *ECCV 2008, Part III. LNCS*, vol. 5304, pp. 668–680. Springer, Heidelberg (2008)
16. Vincent, L.: *Graphs and Mathematical Morphology*. *Signal Processing* 16, 365–388 (1989)
17. Voloshin, V.I.: *Introduction to Graph and Hypergraph Theory*. Nova Science, Bombay (2009)
18. Yang, M., Yang, Y.: *A hypergraph approach to linear network coding in multicast networks*. *IEEE Transactions on Parallel and Distributed Systems* 21, 968–982 (2010)
19. Zhen, L., Jiang, Z.: *Hy-SN: Hyper-graph based semantic network*. *Knowledge-Based Systems* 23(8), 809–816 (2010)

Some Morphological Operators on Simplicial Complex Spaces

Fábio Dias, Jean Cousty, and Laurent Najman

Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge,
Équipe A3SI, ESIEE
{diasf,j.cousty,l.najman}@esiee.fr

Abstract. In this work, we propose a framework that allows to build morphological operators for processing and filtering objects defined on (abstract) simplicial complex spaces. We illustrate with applications to mesh and image processing, for which, on the provided examples, the proposed approach outperforms the classical one.

1 Introduction

Introduced by Poincaré [11] for studying the topology of spaces of arbitrary dimensions, a simplicial complex can be seen as a mesh, *i.e.* a space with a triangulation. The basic building block of the complex is the cell, which can be thought of as a set of elements having various dimensions glued together according to certain rules (*e.g.*, a triangle, its edges and vertices). Although simplicial complexes have a wide variety of different usages (*e.g.*, in computer graphics, in Computer-Aided Design or in modeling), their processing has mostly been considered in term of simplification, for example to obtain a simpler model with less details. However, it is more and more frequent to have data associated with the elements of a mesh (*e.g.* a curvature or a texture). Processing (filtering) the values associated with a mesh is not a common problem *per se* in the literature. On the other hand, filtering is a common theme in image processing, and abstract (simplicial or cubical) complexes have been promoted, in particular by Kovalevsky [7], in order to provide a sound topological basis for image analysis, and are more and more popular [4,11,2]. Then again, the values are most of the time located on one of the elements of the cell, usually the facet, *i.e.* the largest element of the cell. In a purely discrete perspective, being able to deal with smaller elements of the cell will allow a kind of “subpixelic” processing.

In this perspective, mathematical morphology provides a useful toolbox made of non-linear operators. Thanks to their algebraic definitions in the framework of lattices, those morphological operators can be applied to many kinds of organized information, and in particular to simplicial complexes.

The complexes can be considered as a natural generalization of the graphs in the sense that a (symmetric) graph is a one dimensional complex. In the past, several authors studied morphological operators on graphs [14,5,9,3,13]. However, to the best of our knowledge, very few studies exist about basic morphological operators on complexes [8], and none deal with the filtering problem. The goal of this

paper is to help bridging this gap. Our main result is a framework for building morphological operators on complex spaces. As main examples of application, we present a set of operators (erosions/dilations, granulometries/anti-granulometry, and alternate sequential filters) that act on the subcomplexes of a space which is itself a simplicial complex. Although this work is settled in the framework of simplicial complexes, all the results extend to cubical complexes.

The article is organized as follows. Section 2 presents the working space, simplicial complexes and lattices. Then, Section 3 introduces operators acting on the defined spaces and shows that these operators are dilations, identifies their adjoint erosions, and presents morphological operators on subcomplexes resulting from the composition of these adjoint operators. Finally, Section 4 introduces a framework that allows morphological operators depending on dimension parameters to be defined and illustrates their use to image and mesh processing.

2 Lattice of Simplicial Complexes

The goal of this work is to explore mathematical morphology on simplicial complex spaces. To this end, this first section recalls the definition of (abstract) simplicial complexes. Then, after a reminder on lattices, we state that the set of all subcomplexes of the space is a lattice. Hence, as will show the next sections of the paper, morphological operators acting on subcomplexes can be studied.

We call (*abstract*) *simplex* any finite nonempty set. The *dimension* of a simplex x , denoted by $dim(x)$, is the number of its elements minus one. A simplex of dimension n is also called an *n-simplex*.

Fig. 1(a) (resp. b, and c) graphically represents a simplex $x = \{a\}$ (resp. $y = \{a, b\}$ and $z = \{a, b, c\}$) of dimension 0 (resp. 1, 2). Fig. 1(d) shows a set of simplices composed of one 2-simplex ($\{a, b, c\}$), three 1-simplices ($\{a, b\}$, $\{b, c\}$ and $\{a, c\}$) and three 0-simplices ($\{a\}$, $\{b\}$ and $\{c\}$).

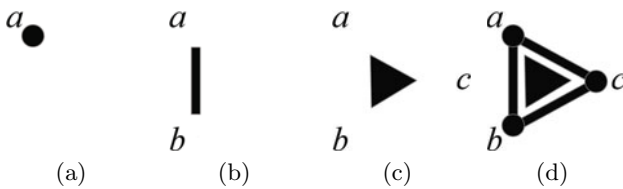


Fig. 1. Graphical representation of (a) a 0-simplex, (b) a 1-simplex, (c) a 2-simplex, and (d) a 2-cell

We call *simplicial complex*, or simply *complex*, any set X of simplices such that, for any $x \in X$, any nonempty subset of x also belongs to X . The *dimension* of a complex is equal to the greatest dimension of its simplices. In the following, a complex of dimension n is also called an *n-complex*.

For instance, Fig. 1(d) represents an elementary 2-complex. The set of black and gray elements in Fig. 2(a) represents a 2-complex made of more simplices.

Important Notations. In this work, the symbol \mathbb{C} denotes a nonempty n -complex, with $n \in \mathbb{N}$. This complex \mathbb{C} stands for our working space in this paper. The set of all subsets of \mathbb{C} is denoted by $\mathcal{P}(\mathbb{C})$.

Recall that a (complete) *lattice* is a partially ordered set in which any two elements have a unique least upper bound, called *supremum*, and a unique greatest lower bound, called *infimum*. Such structures are of particular importance for mathematical morphology. Note that the set $\mathcal{P}(\mathbb{C})$, equipped with the inclusion relation, is a lattice. In this lattice $\mathcal{P}(\mathbb{C})$, the supremum and the infimum are respectively the union and the intersection.

Any subset of \mathbb{C} that is also a complex is called a *subcomplex (of \mathbb{C})*. We denote by \mathcal{C} the set of all subcomplexes of \mathbb{C} . The set \mathcal{C} equipped with the inclusion relation, is a *sublattice* of $\mathcal{P}(\mathbb{C})$ since \mathcal{C} is a subset of $\mathcal{P}(\mathbb{C})$ closed under union and intersection.

A subcomplex X of \mathbb{C} is called a *cell of \mathbb{C}* if there exists a simplex x in X such that X is exactly the set of all subsets of x . Any subcomplex $X \in \mathcal{C}$ is *sup-generated* by the family \mathcal{G} of all cells of \mathbb{C} that are included in X : $X = \cup\{Y \in \mathcal{G}\}$. Conversely, any family \mathcal{G} of cells sup-generates an element of \mathcal{C} . In this sense, the cells can be seen as the elementary building blocks of the complexes.

If X is a subset of \mathbb{C} , we denote by \overline{X} the *complement* of X (in \mathbb{C}): $\overline{X} = \mathbb{C} \setminus X$.

The complement of a subcomplex of \mathbb{C} , in general, is not a subcomplex. For instance, observe in Fig. 2(a) that, if \mathbb{C} is the complex represented in black and gray, then the complement of the gray subcomplex X is not a subcomplex. Thus, contrarily to the lattice $\mathcal{P}(\mathbb{C})$ which is complemented, the lattice \mathcal{C} is not.

Any subset X of \mathbb{C} whose complement \overline{X} is a subcomplex is called a *star (in \mathbb{C})*. We denote by \mathcal{S} the set of all stars in \mathbb{C} . As \mathcal{C} , \mathcal{S} is a sublattice of $\mathcal{P}(\mathbb{C})$.

Note that the intersection $\mathcal{C} \cap \mathcal{S}$ is nonempty since it always contains \emptyset and \mathbb{C} .

The lattice \mathcal{C} is our main working space in the rest of the paper. As a conclusion to this section, let us summarize its properties as follows.

Property 1. *The set \mathcal{C} of the subcomplexes of \mathbb{C} is a complete lattice sup-generated by the set of all cells of \mathbb{C} ; this lattice is not complemented.*

3 Morphological Operators on Simplicial Complex Spaces

Our goal is to investigate morphological dilations and erosions that act on complexes, (where both the inputs and the outputs of the operators are complexes), and that produce nontrivial granulometries, (*i.e.*, granulometries where the dilations are not idempotent). Indeed, such nontrivial granulometries are known to be important in mathematical morphology for analyzing and filtering digital objects according to their size. After a short reminder on morphological adjunctions in the framework of lattices, we present operators that are classical for handling topological spaces such as simplicial complexes. Then, we show that dilations, erosions and granulometries satisfying the above-mentioned properties can be obtained by carefully composing these topological operators.

In mathematical morphology (see, *e.g.*, [12]), any operator that associates elements of a lattice \mathcal{L}_1 to elements of a lattice \mathcal{L}_2 is called a *dilation* if it commutes

with the supremum. Similarly, an operator that commutes with the infimum is called an *erosion*. The notion of adjunction, recalled below, allows dilations and erosions to be classified into pairs of operators leading to granulometries.

Let \mathcal{L}_1 and \mathcal{L}_2 be two lattices whose order relations and suprema are denoted by \leq_1, \leq_2, \vee_1 , and \vee_2 . Two operators $\alpha : \mathcal{L}_2 \rightarrow \mathcal{L}_1$ and $\alpha^A : \mathcal{L}_1 \rightarrow \mathcal{L}_2$ form an *adjunction* (α^A, α) if $\alpha(a) \leq_1 b \Leftrightarrow a \leq_2 \alpha^A(b)$ for every element a in \mathcal{L}_2 and b in \mathcal{L}_1 . It is well known (see, e.g., [12]) that, given two operators α and α^A , if the pair (α^A, α) is an adjunction, then α^A is an erosion and α is a dilation. Furthermore, if α is a dilation, the following relation characterizes its *adjoint erosion* α^A :

$$\forall a \in \mathcal{L}_1, \alpha^A(a) = \vee_2 \{b \in \mathcal{L}_2 \mid \alpha(b) \leq_1 a\} \tag{1}$$

Let us now present two pairs of adjoint operators, which are classical in topology, and that will serve us to obtain nontrivial granulometries on complexes. Let x be a simplex in \mathbb{C} , we set $\hat{x} = \{y \mid y \subseteq x, y \neq \emptyset\}$ and $\tilde{x} = \{y \in \mathbb{C} \mid x \subseteq y\}$.

The operators $Cl : \mathcal{P}(\mathbb{C}) \rightarrow \mathcal{P}(\mathbb{C})$ and $St : \mathcal{P}(\mathbb{C}) \rightarrow \mathcal{P}(\mathbb{C})$ are defined by:

$$\forall X \in \mathcal{P}(\mathbb{C}), Cl(X) = \bigcup \{\hat{x} \mid x \in X\}; \text{ and} \tag{2}$$

$$\forall X \in \mathcal{P}(\mathbb{C}), St(X) = \bigcup \{\tilde{x} \mid x \in X\}. \tag{3}$$

By definition, the operators Cl and St commute under union. Thus, they are dilations on $\mathcal{P}(\mathbb{C})$. Hence, by direct application of Eq. (1), the adjoint erosions Cl^A and St^A of Cl and St are given by:

$$\forall X \in \mathcal{P}(\mathbb{C}), Cl^A(X) = \bigcup \{Y \in \mathcal{P}(\mathbb{C}) \mid Cl(Y) \subseteq X\}; \text{ and} \tag{4}$$

$$\forall X \in \mathcal{P}(\mathbb{C}), St^A(X) = \bigcup \{Y \in \mathcal{P}(\mathbb{C}) \mid St(Y) \subseteq X\}. \tag{5}$$

The four operators presented above are illustrated in Fig. 2, where the subsets X, Y, Z, V , and W , made of gray simplices in Figs. 2(a), 2(b), 2(c), 2(d), and 2(e), satisfy the following relations $Y = St(X)$, $Z = St^A(X)$, $V = Cl(Y)$, $W = Cl^A(Z)$.

Let $X \in \mathcal{P}(\mathbb{C})$. The set $Cl(X)$ (resp. $St(X)$) is the smallest complex (resp. star) that contains X , and the set $Cl^A(X)$ (resp. $St^A(X)$) is the largest complex (resp. star) contained in X . Hence, clearly, \mathcal{C} (resp. \mathcal{S}) is the invariance domain of Cl and Cl^A (resp. St and St^A): $\mathcal{C} = \{X \in \mathcal{P}(\mathbb{C}) \mid Cl(X) = X\} = \{X \in \mathcal{P}(\mathbb{C}) \mid Cl^A(X) = X\}$ (resp. $\mathcal{S} = \{X \in \mathcal{P}(\mathbb{C}) \mid St(X) = X\} = \{X \in \mathcal{P}(\mathbb{C}) \mid St^A(X) = X\}$). These facts are well known in the context of topological spaces [6] where the sets $St(X)$, $St(X)$, $Cl^A(X)$, and $St^A(X)$ are called respectively the (*simplicial*) *closure*, the *star*, the *core*, and the *interior of X*.

Since the operators Cl and St are dilations, they constitute a straightforward choice to investigate morphology on complexes. However, these dilations are idempotent: $Cl \circ Cl(X) = Cl(X)$ and $St \circ St(X) = St(X)$. Thus, they lead to trivial granulometries. In order to obtain nontrivial granulometries, one could consider the composition $Dil = Cl \circ St$. Indeed, the operator Dil is a dilation (since it is a composition of dilations), which, in general, is not idempotent, and whose results are always complexes. By the theorem of composition of adjunctions (see [12], p. 59), the adjoint erosion Er of Dil is given

by $Er = Dil^A = St^A \circ Cl^A$. Due to the remarks of the previous paragraph, the set $Er(X)$ is always a star. Thus, in general, the set $Er(X)$ is not a complex. Hence, the pair (Er, Dil) does not lead to granulometries acting on complexes.

In order to obtain nontrivial granulometries on complexes, let us restrict the operators of Eqs. 2 and 3. More precisely, we define the operators $\diamond : \mathcal{S} \rightarrow \mathcal{C}$ and $\star : \mathcal{C} \rightarrow \mathcal{S}$ by:

$$\forall X \in \mathcal{S}, \diamond(X) = Cl(X); \text{ and} \tag{6}$$

$$\forall Y \in \mathcal{C}, \star(Y) = St(Y). \tag{7}$$

The only differences between \diamond and Cl are the domains of activity of the operators. A similar remark holds true for \star and St . These operators \diamond and \star are also obviously two dilations. Then, using again Eq. 1, the adjoint erosions \diamond^A and \star^A of \diamond and \star are given by:

$$\forall X \in \mathcal{C}, \diamond^A(X) = \bigcup \{Y \in \mathcal{S} \mid \diamond(Y) \subseteq X\}; \text{ and} \tag{8}$$

$$\forall Y \in \mathcal{S}, \star^A(Y) = \bigcup \{X \in \mathcal{C} \mid \star(X) \subseteq Y\}. \tag{9}$$

It can be easily seen that the star $\diamond^A(X)$ is the interior of the complex X and that the complex $\star^A(Y)$ is the core of the star Y . Therefore, one straightforwardly deduces the following property that links the adjoint of \star , St , \diamond , and Cl in a surprising way.

Property 2. *The two following propositions hold true:*

$$\forall X \in \mathcal{C}, \diamond^A(X) = St^A(X); \text{ and} \tag{10}$$

$$\forall Y \in \mathcal{S}, \star^A(Y) = Cl^A(Y). \tag{11}$$

It is known in topology that the closure and interior operators are dual with respect to the complement. Thus, we deduce the following result.

Property 3. *The operators \diamond and \diamond^A (resp. \star and \star^A) are dual w.r.t. the complement in $\mathcal{P}(\mathbb{C})$: we have $\diamond^A(X) = \overline{\diamond(\overline{X})}$, for any $X \in \mathcal{C}$ (resp. $\star^A(Y) = \star(\overline{Y})$, for any $Y \in \mathcal{S}$)*

Note that using directly Eqs. 8 and 9, computing $\diamond^A(X)$ (resp. $\star^A(X)$) requires an exponential time since the family of all stars (complexes) must be considered. On the other hand, as the operators Cl and St are locally defined, $\diamond(X)$ and $\star(X)$ can be computed in linear-time. Hence, as a consequence of Property 3, $\diamond^A(X)$ and $\star^A(X)$ can also be computed in linear-time.

Let us now compose the dilations \diamond and \star , as well as their adjoints, to obtain a pair of adjoint dilations and erosions that act on complexes.

Definition 4. *We define the operators δ and ε acting on \mathcal{C} by:*

$$\delta = \diamond \circ \star \tag{12}$$

$$\varepsilon = \star^A \circ \diamond^A \tag{13}$$

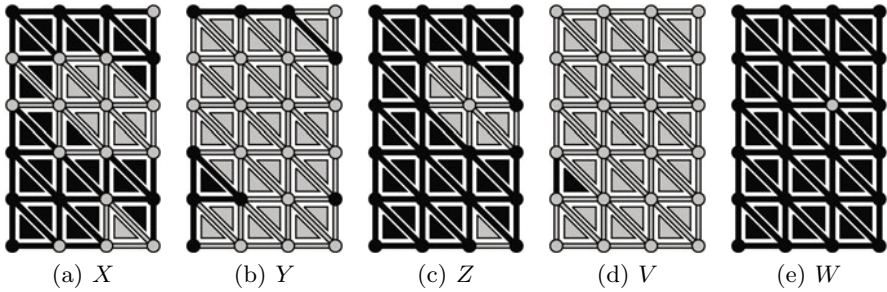


Fig. 2. Illustration of morphological dilations and erosions on complexes [see text]

For instance, Figs. 2(d) and 2(e) represent, in gray, the complexes $V = \delta(X)$ and $W = \varepsilon(X)$, if X is the complex represented in gray in Fig. 2(a).

Due to the theorem of composition of adjunctions (see, e.g., [12], p. 59), the following result can be deduced.

Theorem 5. *The two operators δ and ε are respectively a dilation and an erosion acting on \mathcal{C} . Furthermore, the pair (ε, δ) is an adjunction.*

The dilation δ and the erosion ε , as well as the classical dilations and erosions on grid points by symmetrical structuring elements, are in general not idempotent. However, contrarily to classical dilations and erosions on grid points, the erosion ε and the dilation δ are not dual with respect to the complement since the lattice \mathcal{C} is not complemented.

As the pair (ε, δ) is an adjunction, the compositions $\phi = \varepsilon \circ \delta$ and $\gamma = \delta \circ \varepsilon$ are respectively a *closing* and an *opening*. In other words, the operators ϕ and γ are both increasing and idempotent, whereas ϕ is extensive and γ is anti-extensive. Furthermore, since δ and ε are not idempotent, the operators obtained by iterating ε and δ lead to nontrivial granulometries, as we will detail in the next section.

Figs. 3(b) and 3(d) depict, in gray, the results of ϕ and γ applied to the gray complexes X of Fig. 3(a) and Y of Fig. 3(c). Observe, in particular, that these operators can be intuitively regarded as elementary closing and opening on complexes.

4 Dimensional Operators

From their very definition, simplicial complex spaces allow for handling objects of different dimension (e.g., “curvilinear”, “surfacic” or “volumic” objects), as well as objects of heterogeneous dimension (e.g., made of “curvilinear”, “surfacic” and “volumic” sub-objects). The operators introduced in the previous section add or remove simplices independently of the dimension of the objects. In this section, we introduce a framework for morphological operators that take dimension into account. We first present (Definition 6) the building blocks of the

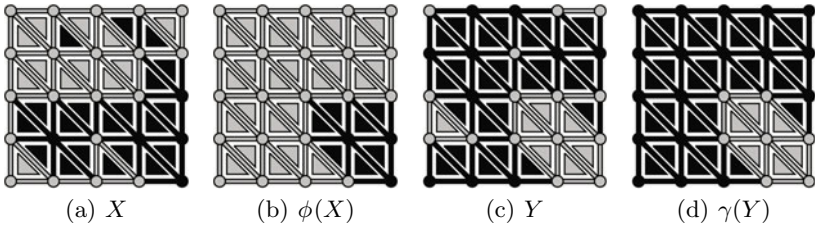


Fig. 3. Illustration of morphological closing and opening on complexes

framework: a family of adjoint operators that depend on dimensional parameters. By combining these building blocks, many morphological operators acting on subcomplexes, substars, and subsets of the space \mathbb{C} can be obtained. As an example, we instantiate (Definition 9) a granulometry and an alternate sequential filter acting on the lattice of all subcomplexes of \mathbb{C} . Then, it is shown that these filters lead to interesting results on images, as suggested by a comparison with classical morphological image filters. Finally, the ability of the proposed filters to smooth subsets of a mesh is also illustrated.

Let $X \subseteq \mathbb{C}$ and let $i \in [0, n]$, we denote by X_i the set of all i -simplices of X : $X_i = \{x \in X \mid \dim(x) = i\}$. In particular, \mathbb{C}_i is the set of all i -simplices of \mathbb{C} . We denote by $\mathcal{P}(\mathbb{C}_i)$ the set of all subsets of \mathbb{C}_i .

Definition 6 (dimensional operators). Let $i, j \in \mathbb{N}$ such that $i \leq j \leq n$. We define the operators $\delta_{i,j}^+$ and $\varepsilon_{i,j}^+$ acting from $\mathcal{P}(\mathbb{C}_i)$ into $\mathcal{P}(\mathbb{C}_j)$ and the operators $\delta_{j,i}^-$ and $\varepsilon_{j,i}^-$ acting from $\mathcal{P}(\mathbb{C}_j)$ into $\mathcal{P}(\mathbb{C}_i)$ as follows:

$\mathcal{P}(\mathbb{C}_i) \rightarrow \mathcal{P}(\mathbb{C}_j)$	$\mathcal{P}(\mathbb{C}_j) \rightarrow \mathcal{P}(\mathbb{C}_i)$
$X \rightarrow \delta_{i,j}^+(X)$ such that $\delta_{i,j}^+(X) = \{x \in \mathbb{C}_j \mid \exists y \in X, y \subseteq x\}$	$X \rightarrow \delta_{j,i}^-(X)$ such that $\delta_{j,i}^-(X) = \{x \in \mathbb{C}_i \mid \exists y \in X, x \subseteq y\}$
$X \rightarrow \varepsilon_{i,j}^+(X)$ such that $\varepsilon_{i,j}^+(X) = \{x \in \mathbb{C}_j \mid \forall y \in \mathbb{C}_i, y \subseteq x \implies y \in X\}$	$X \rightarrow \varepsilon_{j,i}^-(X)$ such that $\varepsilon_{j,i}^-(X) = \{x \in \mathbb{C}_i \mid \forall y \in \mathbb{C}_j, x \subseteq y \implies y \in X\}$

In other words, $\delta_{i,j}^+(X)$ is the set of all j -simplices of \mathbb{C} that include an i -simplex of X , $\delta_{j,i}^-(X)$ is the set of all i -simplices of \mathbb{C} that are included in a j -simplex of X , $\varepsilon_{i,j}^+(X)$ is the set of all j -simplices of \mathbb{C} whose subsets of dimension i all belong to X , and $\varepsilon_{j,i}^-(X)$ is the set of all i -simplices of \mathbb{C} that are not contained in any j -simplex of \overline{X} .

It is interesting to remark that, since a graph is a 1-complex, the operators $\delta_{0,1}^+$, $\delta_{1,0}^-$, $\varepsilon_{0,1}^+$ and $\varepsilon_{1,0}^-$ are exactly the operators δ^\times , δ^\bullet , ε^\times and ε^\bullet , presented by Cousty *et al.* in [3]. These operators are the basic blocks of all morphological graph operators studied in [3]. Thus, the present framework encompasses the one of [3] that itself allows for recovering most graph operators from Vincent and Heijmans [14,5], the operators of Meyer and Angulo [9], and the classical operators by symmetric structuring elements on grid points.

The next property establishes that the dimensional operators of Definition 6 can be classified into pairs of adjoint operators. Thus, as we shall see later, they can be used as building blocks for the design of morphological operators acting on subsets of \mathbb{C} such as complexes or stars.

Property 7 (adjunction and duality). *Let $i, j \in \mathbb{N}$ such that $i \leq j \leq n$.*

- *The pairs $(\varepsilon_{i,j}^+, \delta_{j,i}^-)$ and $(\varepsilon_{j,i}^-, \delta_{i,j}^+)$ are adjunctions.*
- *The operators $\delta_{i,j}^+$ and $\varepsilon_{i,j}^+$ (resp. $\delta_{j,i}^-$ and $\varepsilon_{j,i}^-$) are dual of each other: $\forall X \subseteq \mathbb{C}_i$, $\varepsilon_{i,j}^+(X) = \mathbb{C}_j \setminus \delta_{i,j}^+(\mathbb{C}_i \setminus X)$ (resp. $\forall X \subseteq \mathbb{C}_j$, $\varepsilon_{j,i}^-(X) = \mathbb{C}_i \setminus \delta_{j,i}^-(\mathbb{C}_j \setminus X)$).*

The operators of Section 3 can all be characterized through dimensional operators since, for any X in $\mathcal{P}(\mathbb{C})$, we have $Cl(X) = \bigcup \{ \delta_{j,i}^-(X_j) \mid i, j \in \mathbb{N}, i \leq j \}$, and $St(X) = \bigcup \{ \delta_{i,j}^+(X_i) \mid i, j \in \mathbb{N}, i \leq j \}$.

In fact, by compositions, suprema, and infima of the dimensional operators, many erosions, dilations, openings and closings, which act on \mathcal{C} , \mathcal{S} , $\mathcal{P}(\mathbb{C}_i)$ (with $i \in [0, n]$), and $\mathcal{P}(\mathbb{C})$, and which depend of dimensional parameters, can be designed. As a proof of concept, let us introduce a family of filters that act on the lattice of all subcomplexes of \mathbb{C} and that enriches the granulometries obtained from Section 3.

Definition 8. *Let d be an integer such that $0 \leq d \leq n$ and let $X \in \mathcal{C}$. We define the operators $\gamma_{d/(n+1)}$ and $\phi_{d/(n+1)}$ by, for any $X \in \mathcal{C}$.*

$$\gamma_{d/(n+1)}(X) = \bigcup \{ \delta_{j,i}^-(X_j) \mid j \in [d, n], i \in [0, j] \}; \text{ and} \tag{14}$$

$$\phi_{d/(n+1)}(X) = \bigcup \left[\{ X_i \mid i \in [0, n - d - 1] \} \cup \{ \varepsilon_{n-d,j}^+(X_d) \mid j \in [n - d, n] \} \right] \tag{15}$$

For instance Figs. 4(a), 4(b), 4(c) and 4(d) represent in gray the complexes $\phi_{1/3}(X)$, $\phi_{2/3}(X)$, $\gamma_{1/3}(Y)$, and $\gamma_{2/3}(Y)$, where X and Y are the subcomplexes shown in gray in Figs. 3(a) and 3(c).

Let $d \in [0, n]$, and let X be a subcomplex of \mathbb{C} . It can be seen that $\gamma_{d/(n+1)}(X)$ is the union of the cells of dimension greater than or equal to d that are included in X . On the other hand, $\phi_{d/(n+1)}(X)$ is the union of the cells of X and of those of \mathbb{C} whose elements of dimension between 0 and $n - d$ belong to X .

From these characterizations, we can deduce that the operators $\gamma_{d/(n+1)}$ and $\phi_{d/(n+1)}$ are respectively an opening and a closing on \mathcal{C} .

It can also be remarked that the family of operators $\{ \gamma_{d/(n+1)} \mid d \in [0, n] \}$ and $\{ \phi_{d/(n+1)} \mid d \in [0, n] \}$ are ordered with respect to the value of the index d : $\forall X \in \mathcal{C}$, $\forall d_1, d_2 \in [0, n - 1]$, if $d_1 \leq d_2$, then we have $\phi_{d_1/(n+1)}(X) \subseteq \phi_{d_2/(n+1)}(X)$ and $\gamma_{d_2/(n+1)}(X) \subseteq \gamma_{d_1/(n+1)}(X)$. In morphology, such families are called granulometries. For direct applications to real-life problems, these granulometries may appear useless since they contain only a fixed number n of operators whereas the size of the objects to be analyzed can be arbitrarily large. In order to overcome this problem, in the next definition, these granulometries are extended by composing them with the adjunctions (and their iterated version) of Section 3.

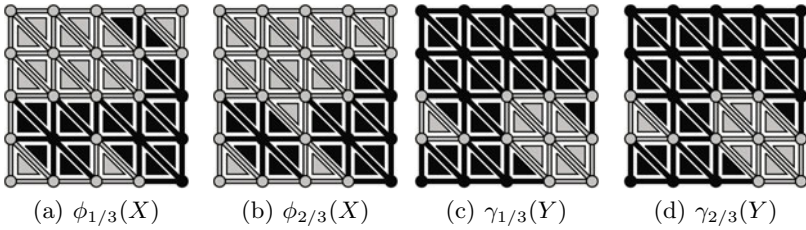


Fig. 4. Dimensional openings and closings applied to the complexes X and Y of Fig. 3

Let α be an operator defined on a lattice \mathcal{L} and let i be a nonnegative integer. The operator α^i is defined by the identity when $i = 0$ and by $\alpha \circ \alpha^{i-1}$ otherwise.

Definition 9 (granulometry, ASF). Let k be a nonnegative integer.

1. We define $\Gamma_{k/(n+1)}$ (resp. $\Phi_{k/(n+1)}$) by $\delta^i \circ \gamma_{d/(n+1)} \circ \varepsilon^i$ (resp. $\varepsilon^i \circ \phi_{d/(n+1)} \circ \delta^i$) where i and d denote respectively the quotient and the remainder of the integer division of k by $(n + 1)$
2. We define $ASF_{k/(n+1)}$ by the identity when $k = 0$ and by $ASF_{k/(n+1)} = \Gamma_{k/n+1} \circ \Phi_{k/n+1} \circ ASF_{(k-1)/(n+1)}$ otherwise.

Property 10. 1. The families $\{\Gamma_{k/(n+1)} | k \in \mathbb{N}\}$ and $\{\Phi_{k/(n+1)} | k \in \mathbb{N}\}$ are granulometries:

- for any $k \in \mathbb{N}$, $\Gamma_{k/(n+1)}$ (resp. $\Phi_{k/(n+1)}$) is an opening (resp. closing) on \mathcal{C} .
 - for any two elements $i, j \in \mathbb{N}$ such that $i \leq j$, we have $\Gamma_{j/(n+1)}(X) \subseteq \Gamma_{i/(n+1)}(X)$ and $\Phi_{i/(n+1)}(X) \subseteq \Phi_{j/(n+1)}(X)$, for any $X \in \mathcal{C}$.
2. The family $\{ASF_{k/(n+1)} | k \in \mathbb{N}\}$ is a family of alternate sequential filters:
- for any two elements $i, j \in \mathbb{N}$, if $i \leq j$, then we have $ASF_{j/(n+1)} \circ ASF_{i/(n+1)} = ASF_{j/(n+1)}$.

Note that if we reduce the granulometric and ASF families of Definition 9 to the operators where k is a multiple of $(n + 1)$, then we recover exactly the granulometries and ASFs induced by the adjunctions of Section 3. In this sense, the proposed granulometries and ASFs enrich the elementary ones on complexes.

Let us now illustrate on an 2D image that, in practice, these operators also enrich the classical morphological filters used in image analysis. To this end , we consider simplicial complexes based on the neighborhood relation given by the hexagonal grid (see e.g., Fig. 2(a)) where the grid points (or pixels) correspond to 0-simplices. In this context, Fig. 5(f) shows the result of $ASF_{6/3}$ applied to the complex derived from the white pixels of image 5(a). For comparison, Fig. 5(c) and Fig. 5(e) show the results of classical alternate sequential filter and graph alternate sequential filters 3 on hexagonal grid. On this image, the ASFs introduced in this paper outperform the classical and graph operators. Nevertheless, they require more iterations (6 vs. 2 and 4). Therefore, in order to compare the proposed ASF with filters using the same number of iterations,

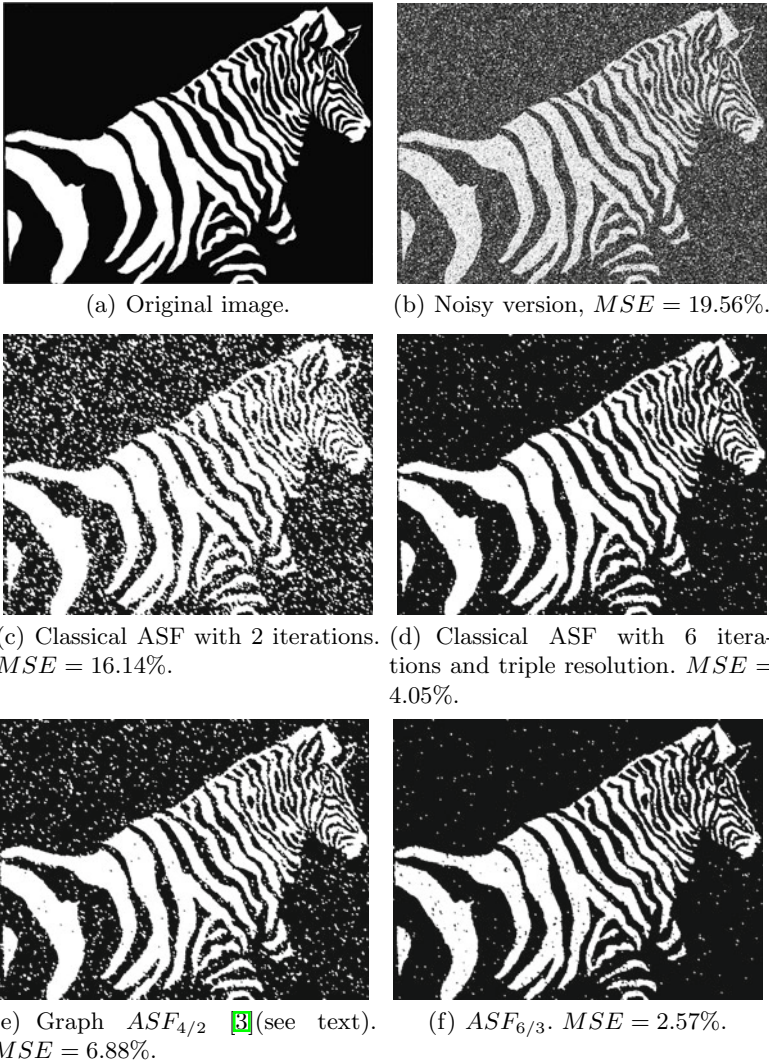


Fig. 5. ASF illustration [see text]

we produce another filtered image (Fig. 5(d)) that is obtained by tripling the resolution of the noisy image and applying a classical ASF of size $2 \times 3 = 6$. It can be seen that this last procedure removes more noise than the classical ASF but does not perform as well as the ASF introduced in the present paper. The mentioned value MSE is the mean square error, multiplied by 100, that is, the percentage of wrong pixels w.r.t. the original image.

Fig. 6(a) shows a rendering of a tridimensional mesh of a statue. Fig. 6(b) shows, in black, the complex resulting from the threshold at level 0.51 of the pseudo-inverse of the mean curvature of that mesh (see 10 for further details

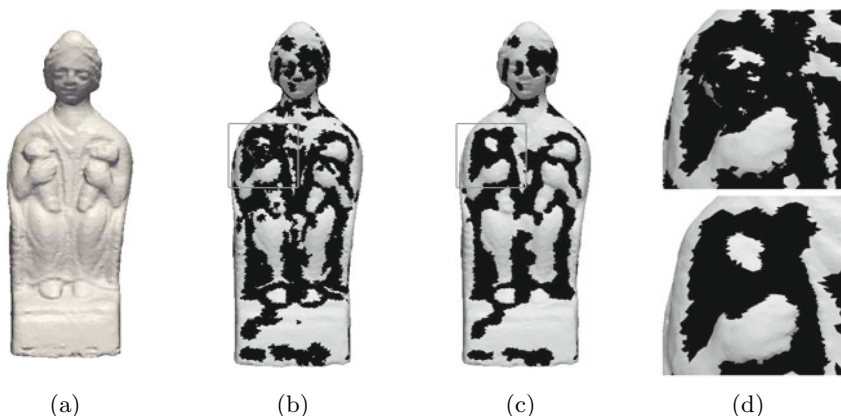


Fig. 6. (a) Rendering of a tridimensional mesh (*i.e.* a 2-complex embedded in \mathbb{R}^3). (b) a threshold of the curvature map of (a). (c) Result of the operator $ASF_{8/3}$ applied to the subcomplex shown in black in (b). (d) Crops, from top to bottom, of (b) and (c) respectively. Data courtesy of the French Museum Center for Research.

and motivations on such procedures). To illustrate the possible smoothing effect of the operator on values associated to a mesh, we present in Fig. [6\(c\)](#) the result of $ASF_{8/3}$ applied to the black subset of the mesh shown on Fig. [6\(b\)](#). Fig. [6\(d\)](#) presents, from top to bottom, zooms on the parts of Fig. [6\(b\)](#) and [6\(c\)](#) that are marked by bold rectangles.

5 Conclusion and Future Work

This paper proposes a framework that allows to build morphological operators for analyzing and filtering objects defined on simplicial complex spaces. In particular, using this framework, we propose a set of operators (erosions/dilations, granulometries, and alternate sequential filters), which act on the lattice of sub-complexes, that are shown to be useful for mesh and image filtering. Furthermore, the proposed framework extends straightforwardly to define operators with similar behavior acting on the lattice of stars of \mathbb{C} . Future work includes a systematic investigation of the morphological operators that can be built based on our framework as well as its straightforward extension to weighted simplicial complexes [\[4,2\]](#). In particular, links with operators from discrete calculus will be highlighted.

Acknowledgement. The authors are grateful to Christian Ronse for his in-depth reading of a previous version of this paper and his numerous helpful comments.

References

1. Couprie, M., Bertrand, G.: New characterizations of simple points in 2D, 3D and 4D discrete spaces. *IEEE Trans. Pattern Analysis and Machine Intelligence* 31(4), 637–648 (2009)
2. Cousty, J., Bertrand, G., Couprie, M., Najman, L.: Collapses and watersheds in pseudomanifolds. In: Wiederhold, P., Barneva, R.P. (eds.) *IWCIA 2009*. LNCS, vol. 5852, pp. 397–410. Springer, Heidelberg (2009)
3. Cousty, J., Najman, L., Serra, J.: Some morphological operators in graph spaces. In: Wilkinson, M.H.F., Roerdink, J.B.T.M. (eds.) *ISMM 2009*. LNCS, vol. 5720, pp. 149–160. Springer, Heidelberg (2009)
4. Grady, L.J., Polimeni, J.R.: *Discrete Calculus*. Springer, Heidelberg (2010)
5. Heijmans, H., Vincent, L.: Graph morphology in image analysis. In: Dougherty, E. (ed.) *Mathematical Morphology in Image Processing*, pp. 171–203. Marcel Dekker, New York (1992)
6. Jänich, K.: *Topology*. Springer, Heidelberg (1984)
7. Kovalevsky, V.A.: Finite topology as applied to image analysis. *CVGIP* 46(2), 141–161 (1989)
8. Loménie, N., Stamon, G.: Morphological mesh filtering and α -objects. *Pattern Recognition Letters* 29(10), 1571–1579 (2008)
9. Meyer, F., Angulo, J.: Micro-viscous morphological operators. In: *ISMM 2007*, pp. 165–176 (2007)
10. Philipp-Foliguet, S., Jordan, M., Najman, L., Cousty, J.: Artwork 3D Model Database Indexing and Classification. *PR* 44(3), 588–597 (2011)
11. Poincaré, H.: Analysis situs. *Journal de l'École Polytechnique* 2(1), 1–178 (1895)
12. Ronse, C., Serra, J.: Algebraic foundations of morphology. In: Najman, L., Talbot, H. (eds.) *Mathematical Morphology: From Theory to Applications*, pp. 35–79. ISTE-Wiley, Chichester (2010)
13. Ta, V.T., Elmoataz, A., Lezoray, O.: Partial difference equations on graphs for mathematical morphology operators over images and manifolds. In: *Procs. of ICIP 2008*, pp. 801–804 (12-15, 2008)
14. Vincent, L.: Graphs and mathematical morphology. *Signal Processing* 16, 365–388 (1989)

Selection of Relevant Nodes from Component-Trees in Linear Time

Nicolas Passat^{1,*} and Benoît Naegel²

¹ Université de Strasbourg, LSIIT, UMR CNRS 7005, France
passat@unistra.fr

² Université Nancy 1, LORIA, UMR CNRS 7503, France
benoit.naegel@loria.fr

Abstract. Component-trees associate to a discrete grey-level image a descriptive data structure induced by the inclusion relation between the binary components obtained at successive level-sets. This article presents a method to extract a subset of the component-tree of an image enabling to fit at best a given binary target selected beforehand in the image. A proof of the algorithmic efficiency of this method is proposed. Application examples related to the extraction of drop caps from ancient documents emphasise the usefulness of this technique in the context of assisted segmentation.

Keywords: component-tree, image analysis, grey-level images.

1 Introduction

The *component-tree* (also known as *dendrone* [2], *confinement tree* [4] or *max-tree* [10]) is a graph-based structure which models some characteristics of a grey-level image by considering its binary level-sets obtained from successive thresholding operations. Component-trees have been involved, in particular, in the development of morphological operators [10,1].

By definition, they are particularly well-suited for the design of methods devoted to process grey-level images based on hypotheses related to the topology (connectedness) and the specific intensity (locally/globally minimal or maximal) of structures of interest. Based on these properties, component-trees have been involved in various kinds of image processing methods [2,4,10,3,5,9,11].

Several works related to component-trees have been devoted to enable their efficient computation [4,10,18]. In particular, the ability to compute them in (quasi-)linear time opens the way to the development of interactive and efficient segmentation methods.

The design of interactive segmentation methods is an active research field (see, e.g., [6] for a recent survey). This dynamism is justified by (i) the increasing necessity to analyse images in a large spectrum of application fields, (ii) the difficulty to develop fully automatic segmentation methods, and (iii) the importance to develop segmentation methods as tools for assisting the user by explicitly using his expertise.

* The research leading to these results has received funding from the French *Agence Nationale de la Recherche* (Grant Agreement ANR-2010-BLAN-0205).

Some of interactive segmentation methods aim at correcting a rough segmentation initially performed in a manual fashion. In this article, we focus on this kind of issue, especially in the (frequent) case where the structures of interest to be segmented are the ones of extremal intensities. In this context, component-trees can be of high usefulness.

Based on these considerations, this article is devoted to answer the following problem: Let I be a grey-level image, let T be its component-tree, and let G be a binary object defined on the same domain as I (assumed to be a rough segmentation of I); how can we determine a part of T (and thus of I) which enables to fit at best G with the lowest computational cost? This “best” approximation can, in particular, be considered from a quantitative point of view, *i.e.*, by finding a solution minimising the amount of false positives/negatives.

The article is organised as follows. In Sec. 2 definitions related to the notion of component-tree are recalled. In Sec. 3 definitions and notations related to the considered issue are proposed. In Sec. 4 some solutions to the proposed problem are described and their linear algorithmic cost is established. Algorithmic considerations are proposed in Sec. 5. An application to the segmentation of drop caps from ancient documents is proposed in Sec. 6 in order to illustrate the relevance of the method and its actual usefulness in real image analysis applications. Conclusions will be found in Sec. 7.

2 Component-Tree

Let $n \in \mathbb{N}^*$. Let us consider an adjacency relation on the discrete grid defined by \mathbb{Z}^n , for instance, the $2n$ - or the $(3^n - 1)$ -adjacency. Let $X \subseteq \mathbb{Z}^n$ be a non-empty set of \mathbb{Z}^n .

We say that two points $x, y \in X$ are connected (in X), and we note $x \sim_X y$, if there exists a sequence $(x_k)_{k=1}^t$ ($t \geq 1$) of elements of X such that $x_1 = x$, $x_t = y$ and x_k, x_{k+1} are adjacent for all $k \in \llbracket 1, t - 1 \rrbracket$. Note that \sim_X is an equivalence relation on X . The connected components of X are the elements of the quotient set X / \sim_X (noted $C[X]$ in the sequel). We say that X is connected if $C[X] = \{X\}$.

Let $E \subset \mathbb{Z}^n$ be a finite connected set. Let $\perp \leq \top \in \mathbb{Z}$ and $V = \llbracket \perp, \top \rrbracket$. A discrete grey-level image I can be defined as a function $I : E \rightarrow V$ (we also note $I \in V^E$).

For any $v \in V$, we define the thresholding function $X_v : V^E \rightarrow \mathcal{P}(E)$ (where $\mathcal{P}(E) = \{Y \mid Y \subseteq E\}$) by $X_v(I) = \{x \in E \mid v \leq I(x)\}$ for all $I \in V^E$.

For any $v \in V$, and any $X \subseteq E$, we define the cylinder function $C_{X,v} : E \rightarrow V$ by $C_{X,v}(x) = v$ if $x \in X$ and \perp otherwise. A discrete image $I \in V^E$ can then be expressed as

$$I = \bigvee_{v \in V} C_{X_v(I),v} = \bigvee_{v \in V} \bigvee_{X \in C[X_v(I)]} C_{X,v} \tag{1}$$

where \bigvee is the pointwise supremum for the sets of functions.

Let $\mathcal{K} = \bigcup_{v \in V} C[X_v(I)]$ be the set of all the connected components obtained from the different thresholdings of I at values $v \in V$. The inclusion relation \subseteq is then a partial order on \mathcal{K} . Let $v_1 \leq v_2 \in V$. Let $B_1, B_2 \subseteq E$ be the binary images defined by $B_k = X_{v_k}(I)$ for $k \in \{1, 2\}$. Let $C_2 \in C[B_2]$ be a connected component of B_2 . Then, there exists a (unique) connected component $C_1 \in C[B_1]$ of B_1 such that $C_2 \subseteq C_1$.

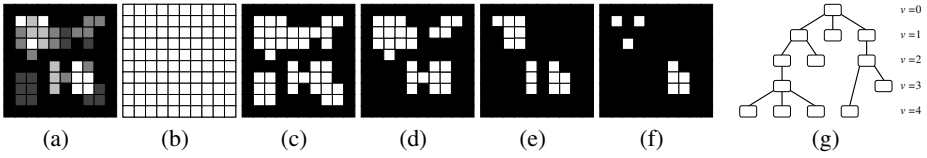


Fig. 1. (a) A grey-level image $I : \llbracket 0, 9 \rrbracket^2 \rightarrow \llbracket 0, 4 \rrbracket$ (from 0, in black, to 4, in white). (b–f) Threshold images $X_\nu(I)$ (white points) for ν varying from 0 (b) to 4 (f). (g) The component-tree of I . Its levels correspond to increasing thresholding values ν . The root (*i.e.*, the upper node located at the level $\nu = 0$) corresponds to the support $E = \llbracket 0, 9 \rrbracket^2$ of the image.

Based on these properties, it can be easily deduced that the Hasse diagram of the partially ordered set (\mathcal{K}, \subseteq) is a tree (*i.e.*, a connected acyclic graph), and more precisely a rooted tree, the root of which is the supremum $X_\perp(I) = E$. This tree is called the *component-tree* of I .

Definition 1. Let $I \in V^E$ be a grey-level image. The component-tree of I is the rooted tree $T = (\mathcal{K}, L, R)$ such that:

- (i) $\mathcal{K} = \bigcup_{\nu \in V} C[X_\nu(I)]$
- (ii) $L = \{(X, Y) \in \mathcal{K}^2 \mid Y \subset X \wedge \forall Z \in \mathcal{K}, Y \subseteq Z \subset X \Rightarrow Y = Z\}$
- (iii) $R = \max(\mathcal{K}, \subseteq) = X_\perp(I) = E$

The elements of \mathcal{K} (*resp.* of L) are the nodes (*resp.* the edges) of T . The element R is the root of T . For any $N \in \mathcal{K}$, we set $ch(N) = \{N' \in \mathcal{K} \mid (N, N') \in L\}$; $ch(N)$ is the set of the children of the node N in T . An example of component-tree is illustrated in Fig. 1.

Each node of \mathcal{K} is a binary connected component distinct from all the other nodes. However, such a connected component can be an element of $C[X_\nu(I)]$ for several (successive) values $\nu \in V$. For each $X \in \mathcal{K}$, we set $m(X) = \max\{\nu \in V \mid X \in C[X_\nu(I)]\} = \min_{x \in X} \{I(x)\}$. We then consider that X is “associated” to the value $m(X)$, *i.e.*, to the highest value of V which generates this connected component.

The following definition, establishing “correlation scores” between a node and a given binary object, will be useful in the sequel of the article.

Definition 2. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $N \in \mathcal{K}$ be a node of T . Let $G \subseteq \mathbb{Z}^n$ be a binary object. We set $p^*(N, G) = |(N \setminus \bigcup_{N' \in ch(N)} N') \cap G|$, and $n(N, G) = |N \setminus G|$. The value $n(N, G)$ is the number of points of N which do not belong to G . The value $p^*(N, G)$ is the number of points of N which belong to G and which do not belong to any children of N .

Remark 3. When building the component-tree of I , it is possible to store, at each node $N \in \mathcal{K}$, the set of points $E_N = N \setminus \bigcup_{N' \in ch(N)} N'$. This leads, in particular, to an algorithmically useful partition $\{E_N\}_{N \in \mathcal{K}}$ of E . In such conditions, for a given binary object $G \subseteq \mathbb{Z}^n$, the computation of all the $p^*(N, G)$ and $n(N, G)$ ($N \in \mathcal{K}$) can obviously be performed in linear time $O(|E|)$. In the sequel, we will assume that $p^*(N, G)$ and $n(N, G)$ have been computed and are then available for every node $N \in \mathcal{K}$.

3 Purpose

Component-trees can be used to develop image processing/analysis procedures based on filtering or segmentation strategies. Such procedures generally consist of determining a subset $\widehat{\mathcal{K}} \subseteq \mathcal{K}$ among the nodes of the component-tree $T = (\mathcal{K}, L, R)$ of a considered image $I : E \rightarrow V$.

When performing segmentation, the (binary) resulting image $I_s \subseteq E$ is defined as the union of the nodes of $\widehat{\mathcal{K}}$, *i.e.*, as

$$I_s = \bigcup_{N \in \widehat{\mathcal{K}}} N \tag{2}$$

In this context, the determination of the nodes to preserve is a complex issue, which can be handled by considering attributes (*i.e.*, qualitative or quantitative information related to each node) to characterise the nodes of interest. An alternative solution is to search the set of nodes $\widehat{\mathcal{K}} \subseteq \mathcal{K}$ which enables to generate a binary object being as similar as possible to a given binary target (*e.g.*, an approximate segmentation obtained from a manual process). In the sequel of the article, we focus on this specific issue, which can be formalised as an optimisation problem.

Problem to solve. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $G \subseteq E$ be a binary image. Let d be a (pseudo-)distance on $\mathcal{P}(E)$. How can we compute a set of nodes $\widehat{\mathcal{K}} \subseteq \mathcal{K}$ such that $d(\bigcup_{N \in \widehat{\mathcal{K}}} N, G)$ is minimal, *i.e.*, such that the best binary object which can be built from \mathcal{K} is as close as possible to G ? More formally, the problem can be summarised as a minimisation problem, consisting of determining

$$\widehat{\mathcal{K}} = \arg \min_{\mathcal{K}' \in \mathcal{P}(\mathcal{K})} \{d(\bigcup_{N \in \mathcal{K}'} N, G)\} \tag{3}$$

An intuitive solution for determining a useful (pseudo-)distance d is to consider the amount of false positives/negatives induced by $\bigcup_{N \in \mathcal{K}'} N$ w.r.t. the considered binary object of interest G .

Definition 4. Let $\alpha \in [0, 1]$. Let $d^\alpha : \mathcal{P}(E) \times \mathcal{P}(E) \rightarrow \mathbb{R}^+$ be the function defined by

$$d^\alpha(X, Y) = \alpha \cdot |X \setminus Y| + (1 - \alpha) \cdot |Y \setminus X| \tag{4}$$

The pseudo-distance d^α constitutes a good similarity criterion between binary objects. Note that $d^0(X, Y) = |Y \setminus X|$ (resp. $d^1(X, Y) = |X \setminus Y|$), *i.e.*, $d^0(X, Y)$ (resp. $d^1(X, Y)$) is the amount of false negatives (resp. false positives) in X w.r.t. Y .

In the next sections, we will consider this (pseudo-)distance. It will be established that it leads to algorithmically efficient processes, and satisfactory applicative results.

¹ The function d^α is actually not a distance since $d^\alpha(X, Y) = d^\alpha(Y, X)$ if and only if $\alpha = 1/2$, $d^\alpha(X, Y) = 0 \Leftrightarrow X = Y$ if and only if $\alpha \in]0, 1[$, and d^α does not satisfy, in general, the triangle inequality.

4 Theoretical Study

4.1 Preliminary Properties

The following property directly derives from the definitions of Sec. [2](#).

Property 5. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $N \in ch(E)$. Let $\mathcal{K}_N = \{N' \in \mathcal{K} \mid N' \subseteq N\}$. Let $I_{|N} \in V^N$ be the grey-level image corresponding to the restriction of I to the node N . The Hasse diagram (\mathcal{K}_N, L_N) of the partially ordered set $(\mathcal{K}_N, \subseteq)$ enables to define the component-tree $T_N = (\mathcal{K}_N, L_N, N)$ of $I_{|N}$ which is actually a subtree of T . Note in particular that $\{E\} \cup \{\mathcal{K}_N\}_{N \in ch(E)}$ is a partition of \mathcal{K} , while $\{(E, N)\}_{N \in ch(E)} \cup \{L_N\}_{N \in ch(E)}$ is a partition of L .*

Definition 6. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $x \in E$. We set $\mathcal{K}_x = \{N \in \mathcal{K} \mid x \in N\}$, $\mathcal{K}_x \subseteq \mathcal{K}$ is the subset of all the nodes of \mathcal{K} which contain x .*

Since $E \in \mathcal{K}$, the following property is obvious, while the next one derives from the structure of the component-tree.

Property 7. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $x \in E$. Then, \mathcal{K}_x is non-empty.*

Property 8. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $x \in E$. Then, $(\mathcal{K}_x, \subseteq)$ is a completely ordered set.*

Definition 9. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $\mathcal{G} : \mathcal{P}(\mathcal{K}) \rightarrow \mathcal{P}(E)$ be the function defined by $\mathcal{G}(\mathcal{K}') = \bigcup_{N \in \mathcal{K}'} N$ for all $\mathcal{K}' \subseteq \mathcal{K}$. We set $\mathcal{Q} = \mathcal{G}(\mathcal{P}(\mathcal{K})) = \{\mathcal{G}(\mathcal{K}')\}_{\mathcal{K}' \subseteq \mathcal{K}}$, \mathcal{Q} is the set of all the binary objects which can be generated from the subsets of nodes of \mathcal{K} .*

Although there exist $2^{|\mathcal{K}|}$ distinct subsets \mathcal{K}' of \mathcal{K} , most of these subsets generate a same binary object of E , more formally, we have $|\mathcal{Q}| \leq |\mathcal{P}(\mathcal{K})|$ (and generally $|\mathcal{Q}| \ll |\mathcal{P}(\mathcal{K})|$).

Property 10. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let \mathcal{Q} be the set of the objects which can be generated from the subsets of nodes of \mathcal{K} . Let $Q \in \mathcal{Q}$. Then, we have*

$$C[Q] = \min_{\subseteq} \mathcal{G}^{-1}(Q) \tag{5}$$

Less formally, the set of the connected components of Q is actually a subset of nodes of \mathcal{K} which is included in any other subset of nodes \mathcal{K}' of \mathcal{K} generating Q . Such sets \mathcal{K}' are then redundant (they contain in particular some nodes which are included in other nodes, and then useless for the generation of Q).

4.2 Main Properties

Smallest Superset / Largest Subset. In this subsection, we first focus on a specific case of the considered issue, which consists of finding a subset of nodes of the component-tree of an image I such that the object generated by these nodes is *included in*

(resp. *includes*) the binary target G and is the *largest* (resp. the *smallest*) one verifying this property. This problem is equivalent to consider a pseudo-distance d which only takes into account the amount of false negatives (resp. false positives) w.r.t. G .

The next property establishes that there exists a (unique) solution to this problem.

Property 11. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let Q be the set of the objects which can be generated from the subsets of nodes of \mathcal{K} . Let $G \subseteq E$. Then there exist $G^+, G^- \in Q$ such that*

$$G^+ = \min_{\subseteq} \{Q \in Q \mid G \subseteq Q\} \tag{6}$$

$$G^- = \max_{\subseteq} \{Q \in Q \mid Q \subseteq G\} \tag{7}$$

Proof. If $G = \emptyset$, by setting $G^+ = \emptyset \in Q$, we are done. Let us now suppose that $G \neq \emptyset$. For any $x \in G$, we set $N_x = \min_{\subseteq} \mathcal{K}_x$. Let $G^+ = \bigcup_{x \in G} N_x$. Then we have $G^+ \in Q$ and $G \subseteq G^+$. Let $Q' \in Q$ such that $G \subseteq Q'$. Let $y \in G^+$. If $y \in G$, then we have $y \in Q'$. Let us now suppose that $y \in G^+ \setminus G$. Then, there exists $x \in G$ such that $y \in N_x$. Since $x \in G \subseteq Q'$, there exists $N \in \mathcal{K}_x$ such that $N \subseteq Q'$. But then, we have $y \in N_x \subseteq N \subseteq Q'$. Consequently, we have $Q \subseteq Q'$, and thus $G^+ = \min_{\subseteq} \{Q \in Q \mid G \subseteq Q\}$.

Let $G^- = \bigcup_{N \in \mathcal{K} \wedge N \subseteq G} N$. We have $G^- \in Q$ and $G^- \subseteq G$. Let $Q' \in \{Q \in Q \mid Q \subseteq G\}$. Let us suppose that there exists $x \in Q' \setminus G^-$. In particular, we have $x \in G$. There exists $N_x \in \mathcal{K}_x$ such that $N_x \subseteq Q'$. If $N_x \subseteq G$ then we have $x \in N_x \subseteq G^-$: contradiction. If $N_x \not\subseteq G$ then we have $Q' \not\subseteq G$: contradiction. Consequently, for all $x \in Q'$, we have $x \in G^-$, and thus $G^- = \max_{\subseteq} \{Q \in Q \mid Q \subseteq G\}$. \square

We define now two functions which enable to compute these solutions G^+ and G^- (Def. 12, Props. 13 and 14) and we show that they authorise a computation in linear time w.r.t. the size (*i.e.*, the number of nodes) of the component-tree of the considered image I or the size of the support E of this image (Prop. 15).

Definition 12. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $G \subseteq E$. Let $\mathcal{F}^+, \mathcal{F}^- \in \mathcal{P}(\mathcal{K})^{\mathcal{K}}$ be the functions recursively defined, for all $N \in \mathcal{K}$, by*

$$\mathcal{F}^+(N) = \begin{cases} \{N\} & \text{if } p^*(N, G) \neq 0 \\ \bigcup_{N' \in \text{ch}(N)} \mathcal{F}^+(N') & \text{if } p^*(N, G) = 0 \end{cases} \tag{8}$$

$$\mathcal{F}^-(N) = \begin{cases} \{N\} & \text{if } n(N, G) = 0 \\ \bigcup_{N' \in \text{ch}(N)} \mathcal{F}^-(N') & \text{if } n(N, G) \neq 0 \end{cases} \tag{9}$$

In particular, if $\text{ch}(N) = \emptyset$, we have $\bigcup_{N' \in \text{ch}(N)} \mathcal{F}^-(N') = \bigcup_{N' \in \text{ch}(N)} \mathcal{F}^+(N') = \emptyset$, which guarantees the termination of these recursive definitions. The function \mathcal{F}^+ (resp. \mathcal{F}^-) provides, for any node $N \in \mathcal{K}$, the subset of nodes of the subtree of T having N for root, which enables to generate the set $(G \cap N)^+$ (see Eq. 6) (resp. $(G \cap N)^-$ (see Eq. 7)) for the restriction of the image I to N .

Property 13. *Let $\sigma \in \{+, -\}$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $G \subseteq E$. Then we have*

$$C[G^\sigma] = \mathcal{F}^\sigma(E) \tag{10}$$

Proof. Let $X, Y \in \mathcal{F}^\sigma(E)$. By definition, we have $X, Y \in \mathcal{K}$. Moreover, if $X \neq Y$, it obviously comes that $X \cap Y = \emptyset$. Consequently, there exists $Q \in \mathcal{Q}$ such that $\mathcal{F}^\sigma(E) = C[Q]$. By induction from the definition of $\mathcal{F}^+(E)$ (resp. $\mathcal{F}^-(E)$), we easily deduce that $\bigcup_{N \in \mathcal{F}^+(E)} N = \bigcup_{N \in \mathcal{K} \wedge p^*(N, G) \neq 0} N$ (resp. $\bigcup_{N \in \mathcal{F}^-(E)} N = \bigcup_{N \in \mathcal{K} \wedge n(N, G) = 0} N$). In particular, it follows that $\bigcup_{N \in \mathcal{F}^+(E)} N \in \{Q \in \mathcal{Q} \mid G \subseteq Q\}$ (resp. $\bigcup_{N \in \mathcal{F}^-(E)} N \in \{Q \in \mathcal{Q} \mid Q \subseteq G\}$). Let $N \in \mathcal{F}^+(E)$. Let $y \in G$ such that $y \in N$ and $y \notin \bigcup_{N' \in \text{ch}(N)} N'$ (such a point y exists as $p^*(N, G) \neq 0$). Then, $N = \min_{\subseteq} \mathcal{K}_y$, and since $y \in G^+$, we must have $N \subseteq G^+$. Consequently, we have $\bigcup_{N \in \mathcal{F}^+(E)} N \subseteq G^+$, and then $\bigcup_{N \in \mathcal{F}^+(E)} N = G^+$ and $\mathcal{F}^+(E) = C[G^+]$. Let $x \in G^- \setminus \bigcup_{N \in \mathcal{F}^-(E)} N$. Then, there exists $N \in \mathcal{K}_x$ such that $N \subseteq G^-$. As $x \notin \bigcup_{N \in \mathcal{F}^-(E)} N$, we have $N \notin \mathcal{F}^-(E)$, and in particular, $n(N, G) \neq 0$. But then, there exists $y \in N$ such that $y \notin G$, and thus, $G^- \not\subseteq G$: contradiction. Consequently, we have $G^- \subseteq \bigcup_{N \in \mathcal{F}^-(E)} N$, and then $G^- = \bigcup_{N \in \mathcal{F}^-(E)} N$ and $\mathcal{F}^-(E) = C[G^-]$. \square

The following property immediately derives from Prop. 13.

Property 14. *Let $\sigma \in \{+, -\}$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $G \subseteq E$. Then we have*

$$G^\sigma = \bigcup_{N \in \mathcal{F}^\sigma(E)} N \tag{11}$$

Property 15. *Let $\sigma \in \{+, -\}$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $G \subseteq E$. Then $C[G^\sigma]$ (and thus G^σ) can be computed with a linear algorithmic complexity $O(\max\{|\mathcal{K}|, |E|\})$, w.r.t. the number of nodes of the tree or the size of the image.*

Proof. From the definition of $\mathcal{F}^\sigma(E)$, it is easily proved that each node is processed at most once. For each one of these $O(|\mathcal{K}|)$ processed nodes, one equality (related to $p^*(N, G)$ or $n(N, G)$, which are assumed to be precomputed, see Remark 3) is tested, and the status of the node (“in” or “out of” the result $\mathcal{F}^\sigma(E)$) is possibly modified. These two operations have a constant algorithmic complexity $O(1)$. The whole process then presents a linear complexity $O(|\mathcal{K}|)$. The generation of G^σ from $\mathcal{F}^\sigma(E)$ can be performed by modifying, for each node N of \mathcal{K} and for each point x of N (these points being stored in E_N for each node N , see Remark 3) the status of x to indicate that it belongs to G^σ . This process then presents an algorithmic complexity $O(|E|)$. Hence the result holds. \square

General Case. We now focus on the general case of the problem stated in Sec. 3, which consists of finding a set of nodes $\widehat{\mathcal{K}}$ of the component-tree of an image I verifying Eq. (4), for the pseudo-distance d^α proposed in Def. 4. The purpose is then to find the best compromise (according to a chosen weight $\alpha \in [0, 1]$) between the amount of false positives and false negatives w.r.t. a binary target G .

Since the set Q of the objects which can be generated from the subsets of nodes of a component-tree is finite, there necessarily exists a solution to this problem. Hereafter, we show that such a solution (Def. 16) can be computed in linear time w.r.t. the size (i.e., the number of nodes) of the component-tree of the considered image I or the size of the support E of this image (Props. 19 and 20).

Definition 16. Let $\alpha \in [0, 1]$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let $G \subseteq E$. Let $< \in \{<, \leq\}$. Let $\mathcal{F}^\alpha : \mathcal{K} \rightarrow \mathcal{P}(\mathcal{K})$ and $c^\alpha : \mathcal{K} \rightarrow \mathbb{R}^+$ be the functions recursively cross-defined, for all $N \in \mathcal{K}$, by

$$(\mathcal{F}^\alpha(N), c^\alpha(N)) = \begin{cases} (\{N\}, \alpha.n(N, G)) & \text{if } \alpha.n(N, G) < (1 - \alpha).p^*(N, G) + \sum_{N' \in ch(N)} c^\alpha(N') \\ (\bigcup_{N' \in ch(N)} \mathcal{F}^\alpha(N'), (1 - \alpha).p^*(N, G) + \sum_{N' \in ch(N)} c^\alpha(N')) & \text{otherwise} \end{cases} \quad (12)$$

In particular, if $ch(N) = \emptyset$, we have $\bigcup_{N' \in ch(N)} \mathcal{F}^\alpha(N') = \emptyset$ (which guarantees the termination of these recursive definitions), and $\sum_{N' \in ch(N)} c^\alpha(N') = 0$. The function \mathcal{F}^α provides, for any node $N \in \mathcal{K}$, the subset of nodes of the subtree of T having N for root, which enables to generate the set $(G \cap N)^\alpha$ (see Eq. (13), below) for the restriction of the image I to N . The function c^α provides the cost (w.r.t. d^α) of this best solution.

Definition 17. Let $\alpha \in [0, 1]$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let \mathcal{Q} be the set of the objects which can be generated from the subsets of nodes of \mathcal{K} . Let $G \subseteq E$. We define $G^\alpha \in \mathcal{Q}$ as

$$G^\alpha = \bigcup_{N \in \mathcal{F}^\alpha(E)} N \quad (13)$$

From a reasoning similar to (and actually simpler than) the one of Prop. 13, we have the following result.

Property 18. Let $\alpha \in [0, 1]$. Let $I \in V^E$ be a grey-level image. Let $G \subseteq E$. Then we have

$$\mathcal{F}^\alpha(E) = C[G^\alpha] \quad (14)$$

Property 19. Let $\alpha \in [0, 1]$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I . Let \mathcal{Q} be the set of the objects which can be generated from the subsets of nodes of \mathcal{K} . Let $G \subseteq E$. Then, we have

$$d^\alpha(G^\alpha, G) = c^\alpha(E) = \min_{Q \in \mathcal{Q}} \{d^\alpha(Q, G)\} \quad (15)$$

Proof. Let us suppose that $ch(E) = \emptyset$. Then we have $Q = \{\emptyset, E\}$, $d^\alpha(\emptyset, G) = (1 - \alpha).p(E, G)$ and $d^\alpha(E, G) = \alpha.n(E, G)$. If $\alpha.n(E, G) < (1 - \alpha).p^*(E, G) + \sum_{N \in ch(E)} c^\alpha(N)$, i.e., if $\alpha.n(E, G) < (1 - \alpha).p(E, G)$, then $\mathcal{F}^\alpha(E) = \{E\}$, $c^\alpha(E) = \alpha.n(E)$ and thus we have $d^\alpha(G^\alpha, G) = d^\alpha(E, G) = c^\alpha(E) = \min_{Q \in \mathcal{Q}} \{d^\alpha(Q, G)\}$. If $\alpha.n(E, G) \not< (1 - \alpha).p(E, G)$, then we have $\mathcal{F}^\alpha(E) = \emptyset$, $c^\alpha(E) = (1 - \alpha).p(E, G) = (1 - \alpha).p^*(E, G)$ and thus, $d^\alpha(G^\alpha, G) = d^\alpha(\emptyset, G) = c^\alpha(E) = \min_{Q \in \mathcal{Q}} \{d^\alpha(Q, G)\}$. Consequently, the property is true whenever $ch(E) = \emptyset$. Let us now suppose that $ch(E) \neq \emptyset$ and that the property holds for any $N \in ch(E)$ (w.r.t. $I|_N$, T_N and $G \cap N$, instead of I , T and G , see Prop. 5). Note that $\min_{Q \in \mathcal{Q}} \{d^\alpha(Q, G)\} = \min\{d^\alpha(E, G), \min_{Q \in \mathcal{Q} \setminus \{E\}} \{d^\alpha(Q, G)\}\}$, while $d^\alpha(E, G) = \alpha.|E \setminus G| = \alpha.n(E)$, and $\min_{Q \in \mathcal{Q} \setminus \{E\}} \{d^\alpha(Q, G)\} = \min_{Q \in \mathcal{Q} \setminus \{E\}} \alpha.|Q \setminus G| + (1 - \alpha).|G \setminus Q|$. Note also that $\{Q \cap N\}_{N \in ch(E)}$ is a partition of Q whenever $Q \neq E$ while $\{G \setminus \bigcup_{N \in ch(E)} N\} \cup \{G \cap N\}_{N \in ch(E)}$ is a partition of G (by omitting the possibly empty subsets). If $Q \neq E$, we have

$d^\alpha(Q, G) = \alpha. |Q \setminus G| + (1 - \alpha). |G \setminus Q| = \alpha. | \bigcup_{N \in \text{ch}(E)} (Q \cap N) \setminus G | + (1 - \alpha). | ((G \setminus \bigcup_{N \in \text{ch}(E)} N) \cup \bigcup_{N \in \text{ch}(E)} (G \cap N)) \setminus Q | = \sum_{N \in \text{ch}(E)} \alpha. | (Q \cap N) \setminus G | + (1 - \alpha). | (G \setminus \bigcup_{N \in \text{ch}(E)} N) \setminus Q | + \sum_{N \in \text{ch}(E)} (1 - \alpha). | (G \cap N) \setminus Q | = \sum_{N \in \text{ch}(E)} (\alpha. | (Q \cap N) \setminus (G \cap N) | + (1 - \alpha). | (G \cap N) \setminus (Q \cap N) |) + (1 - \alpha). | G \setminus \bigcup_{N \in \text{ch}(E)} N | = \sum_{N \in \text{ch}(E)} (\alpha. | (Q \cap N) \setminus (G \cap N) | + (1 - \alpha). | (G \cap N) \setminus (Q \cap N) |) + (1 - \alpha). p^*(E)$. From the above partition properties, it then comes that $\min_{Q \in \mathcal{Q} \setminus \{E\}} \{d^\alpha(Q, G)\} = \min_{Q \in \mathcal{Q} \setminus \{E\}} \{ \sum_{N \in \text{ch}(E)} (\alpha. | (Q \cap N) \setminus (G \cap N) | + (1 - \alpha). | (G \cap N) \setminus (Q \cap N) |) + (1 - \alpha). p^*(E) \} = (1 - \alpha). p^*(E) + \sum_{N \in \text{ch}(E)} \min\{\alpha. | (Q \cap N) \setminus (G \cap N) | + (1 - \alpha). | (G \cap N) \setminus (Q \cap N) |\} = (1 - \alpha). p^*(E) + \sum_{N \in \text{ch}(E)} d^\alpha(Q \cap N, G \cap N) = (1 - \alpha). p^*(E) + \sum_{N \in \text{ch}(E)} c^\alpha(N)$, by induction hypothesis. Consequently, $\min_{Q \in \mathcal{Q}} \{d^\alpha(Q, G)\} = \min\{\alpha. n(E), (1 - \alpha). p^*(E) + \sum_{N \in \text{ch}(E)} c^\alpha(N)\}$, and the result follows by induction from Def. [16](#). \square

Property 20. *Let $\alpha \in [0, 1]$. Let $I \in V^E$ be a grey-level image. Let $G \subseteq E$. Then $\mathcal{F}^\alpha(E) = C[G^\alpha]$ (and thus G^α) can be computed with an algorithmic complexity $O(\max\{|\mathcal{K}|, |E|\})$, linear w.r.t. the number of nodes of the tree or the size of the image.*

Proof. The proof is similar to the proof of Prop. [15](#). The only difference lies in the fact that the set of conditions to be tested ($\alpha. n(N) < (1 - \alpha). p^*(N) + \sum_{N' \in \text{ch}(N)} c^\alpha(N')$) requires at most $|\mathcal{K}|$ comparison operations ($<$) and $4. |\mathcal{K}|$ arithmetic operations ($., +, -$), while the computation of all the terms $c^\alpha(\cdot)$ involves (at most) the value $c^\alpha(N')$ only once for any $N' \in \mathcal{K}$, leading to less than $|\mathcal{K}|$ additions in the set of all the \sum terms. Such supplementary operations then do not increase the algorithmic complexity $O(|\mathcal{K}|)$ of the computation of $\mathcal{F}^\alpha(E)$ by comparison to $\mathcal{F}^\sigma(E)$. Hence the result holds. \square

Remark 21. *The set of nodes $\mathcal{F}^\alpha(E)$ and its associated binary object G^α enable to minimise $d^\alpha(\cdot, G)$, and thus to obtain an optimal solution to the issue considered in this work. However, $\mathcal{F}^\alpha(E)$ and G^α are generally not unique. To illustrate this assertion, let us consider the trivial case where $G = \emptyset$ (resp. $G = E$) and $\alpha = 0$ (resp. $\alpha = 1$). Obviously, in such a case, any set of nodes and any associated binary object minimise $d^0(\cdot, G)$ (resp. $d^1(\cdot, G)$), which is always equal to 0. However, the way to define $<$ in Eq. [12](#) enables to break this non-determinism by choosing to favour the smallest ($<$) or the largest (\leq) solution (w.r.t. the inclusion relation \subseteq) among all the possible ones. In particular, if $<$ is set to $<$ (resp. to \leq) we have $\mathcal{F}^+ = \mathcal{F}^0$ (resp. $\mathcal{F}^- = \mathcal{F}^1$) (the easy proof of this assertion is left to the reader).*

5 Algorithmics

From the above study, which provides an answer to the question stated in Sec. [1](#), we can derive the method described in Alg. [1](#) (For the sake of readability, this algorithm, which is intrinsically recursive, is described in an iterative fashion.)

In its general form, the method corresponds to Def. [16](#) which solves the general case considered in Sec. [4.2](#). In the specific case where $\alpha = 0$ and $< = <$ (resp. $\alpha = 1$ and $< = \leq$), the method corresponds to Eq. [8](#) (resp. Eq. [9](#)) in Def. [12](#) which solves the specific case of the smallest result including (resp. the largest result included in) the rough segmentation, considered in Sec. [4.2](#).

Algorithm 1. Segmentation method**Input**

$I : E \rightarrow V$ (image to be segmented);
 $G \subseteq E$ (rough segmentation of I)
 $\alpha \in [0, 1]$ (weight parameter for false positives/negatives)
 $< \in \{<, \leq\}$ (order involved in the cost minimisation formula)

Output

$G^\alpha \subseteq E$ (final segmentation of I)

Algorithm

1 - *Component-tree computation*

$T = (\mathcal{K}, L, R)$ (component-tree of I)

for all $N \in \mathcal{K}$ **do**

$E_N = N \setminus \bigcup_{N' \in \text{ch}(N)} N'$

$p^*(N, G) = |E_N \cap G|$

$n(N, G) = |N \setminus G|$

end for

2 - *Cost minimisation*

for $v = \top$ **to** \perp **do**

for all $N \in \mathcal{K}$ such that $m(N) = v$ **do**

if $\alpha \cdot n(N, G) < (1 - \alpha) \cdot p^*(N, G) + \sum_{N' \in \text{ch}(N)} c^\alpha(N')$ **then**

$c^\alpha(N) = \alpha \cdot n(N, G)$

$\mathcal{F}^\alpha(N) = \{N\}$

else

$c^\alpha(N) = (1 - \alpha) \cdot p^*(N, G) + \sum_{N' \in \text{ch}(N)} c^\alpha(N')$

$\mathcal{F}^\alpha(N) = \bigcup_{N' \in \text{ch}(N)} \mathcal{F}^\alpha(N')$

end if

end for

end for

3 - *Result computation*

$G^\alpha = \bigcup_{N \in \mathcal{F}^\alpha(E)} N$

6 Application Example: Assisted Segmentation of Drop Caps

As established in the previous sections, given an image, a rough segmentation, and a parameter controlling the trade-off between false positives and false negatives, it is possible to compute, in linear time, the best segmentation composed by the connected components stored in the component-tree of the image. Based on Alg. 1, an interactive segmentation software tool has been developed, and applied to the extraction of drop caps textural parts from ancient documents [7]. These drop caps are issued from the Madonne database OLDB (Ornamentals Letters DataBase), which consists of more than 6000 grey-scale graphical decorative initials extracted from archival documents [8]. Drop caps images are composed of a letter (uppercase) part and textural parts. They are noisy and contain artifacts such as superimposed text, coming from neighbouring book pages.

² We would like to thank the *Centre d'Études Supérieures de la Renaissance* for the permission to use their archival documents.

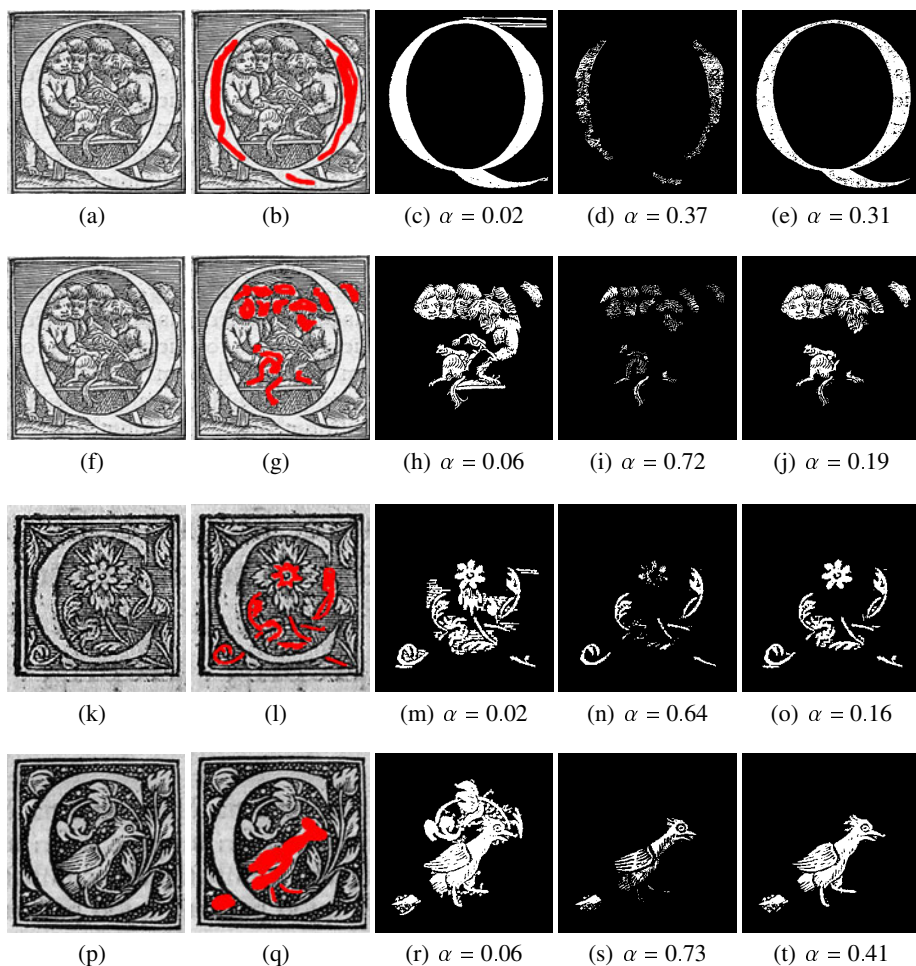


Fig. 2. Segmentation process. First column: initial images. Second column: user-defined rough segmentations (in red). Third to fifth columns: examples of proposed segmentation results for different values of α (fifth column: best obtained segmentation).

The size of these images varies from 150×150 to 750×750 pixels. In this study, we are interested in the fast extraction of objects belonging to the textural (background) part of the drop cap (these objects are of extremal intensity, which is compliant with the requirements of the method). The components are then used afterwards as query objects in a system for drop caps retrieval and indexation.

The segmentation protocol, exemplified in Fig. 2 is the following one. Given a drop cap image, (first column), the user proposes a manually-defined rough segmentation (second column). He can then choose the most satisfactory segmentation by simply interactively tuning the α value (third to fifth column) between 0 and 1 in a threshold-like fashion. For each chosen α , the segmentation is computed on the fly, in real time.

7 Conclusion

In this article, it has been established that the component-tree structure can be used to compute, in linear time, a binary object which fits at best (w.r.t. false positives/negatives criteria) a given binary target which can be assumed to identify structures of interest in a digital grey-level image.

Based on this result a segmentation method has been proposed and successfully applied to the case of document analysis, emphasising the relevance of the approach.

In a further extended version of this work, it will be shown that this interactive segmentation method can be optimised from both time and space point of views, by establishing in particular the increasing property of the results w.r.t. the α values. Additional applications to medical images will also be proposed.

References

1. Breen, E.J., Jones, R.: Attribute openings, thinnings, and granulometries. *Computer Vision and Image Understanding* 64(3), 377–389 (1996)
2. Hanusse, P., Guillaud, P.: Sémantique des images par analyse dendronique. In: RFIA 1991, vol. 2, pp. 577–588 (1991)
3. Jones, R.: Connected filtering and segmentation using component trees. *Computer Vision and Image Understanding* 75(3), 215–228 (1999)
4. Mattes, J., Demongeot, J.: Efficient algorithms to implement the confinement tree. In: Nyström, I., Sanniti di Baja, G., Borgefors, G. (eds.) DGCI 2000. LNCS, vol. 1953, pp. 392–405. Springer, Heidelberg (2000)
5. Mattes, J., Richard, M., Demongeot, J.: Tree representation for image matching and object recognition. In: Bertrand, G., Couprie, M., Perroton, L. (eds.) DGCI 1999. LNCS, vol. 1568, pp. 298–312. Springer, Heidelberg (1999)
6. McGuinness, K., O'Connor, N.E.: A comparative evaluation of interactive segmentation algorithms. *Pattern Recognition* 43(2), 434–444 (2010)
7. Naegel, B., Wendling, L.: Combining shape descriptors and component-tree for recognition of ancient graphical drop caps. In: VISAPP 2009, vol. 2, pp. 297–302 (2009)
8. Najman, L., Couprie, M.: Building the component tree in quasi-linear time. *IEEE Transactions on Image Processing* 15(11), 3531–3539 (2006)
9. Ouzounis, G.K., Wilkinson, M.H.F.: Mask-based second-generation connectivity and attribute filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(6), 990–1004 (2007)
10. Salembier, P., Oliveras, A., Garrido, L.: Anti-extensive connected operators for image and sequence processing. *IEEE Transactions on Image Processing* 7(4), 555–570 (1998)
11. Urbach, E.R., Roerdink, J.B.T.M., Wilkinson, M.H.F.: Connected shape-size pattern spectra for rotation and scale-invariant classification of gray-scale images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(2), 272–285 (2007)

Image Denoising with a Constrained Discrete Total Variation Scale Space

Igor Ciril¹ and Jérôme Darbon²

¹ LMCS, IPSA

² CMLA, ENS Cachan, CNRS, PRES UniverSud

Abstract. This paper describes an approach for performing image restoration using a coupled differential system that both simplifies the image while preserving its contrast. The first process corresponds to a differential inclusion involving discrete Total Variations that simplifies more and more the observed image as time evolves. The second one extracts some pertinent geometric information contained in the series of simplified images and recovers the contrast using Bregman distances. Convergence and exact computational properties of the method rely on the discrete and combinatorial properties of discrete Total Variations.

Keywords: Discrete Total Variation, Bregman Distances, Differential Inclusions, Network Flows.

1 Introduction

Minimization of the Total Variation (TV) with a quadratic data fidelity term is a popular tool for performing image restoration since the seminal work of [3,23]. Although the solution has sharp boundaries it is also known that the minimizer may present a loss of contrast [17,24]. In this paper we propose an approach to address this issue within the framework of a coupled scale-space process.

One popular approach to avoid the loss of contrast consists in considering robust edge-preserving priors [11,5,8,12,18,19,25] instead of TV. Such regularization terms aim at not penalizing too much large gradient that are assumed to correspond to a contour in the reconstructed image. Among these priors, the Potts model and truncated-quadratic prior are the most well-known. However, such a prior yields a non-convex optimization problems where a global minimizer cannot be generally computed in practice. Thus a local minimum or a critical point is obtained using an approximation algorithm [8].

Another approach relies on a Bregman distance procedure that is originally proposed by Osher et al. in [20]. This scheme is an iterative method that consists of minimizing a sequence of convex minimization problems where each of them refines at each step a degraded image. More precisely, the process starts from a constant image and converges toward the observed noisy image. It relies on iterating the following two steps: the normals of the levels are firstly filtered using TV while a surface is approximately fitted to these estimated normal in a second step. These two operations are iterated until the reconstructed image satisfies a

prescribe criterion. The contrast of the reconstructed image is much better than the one obtained from a pure TV approach since it emphasized the geometric information contained in the normals of the level lines as opposed to the actual gray values. This procedure has also received the name of inverse scale space [4] since objects of finer scale are incrementally added to the reconstructed image. Note that this approach can also be rewritten in terms of Bregman distance [20]. In [10], some theoretical links between the iterative Bregman distance scheme and the use of robust edge preserving priors are highlighted.

In this paper we consider an approach that relies on coupling the TV-flow (which corresponds to the solution of a differential inclusion) that incrementally simplifies the original noisy image with a procedure that intends to recover the contrast. More precisely, the flow is used to extract some information about the geometry of the image: in this paper we consider the relative order between any two adjacent pixels (note that this information essentially corresponds to considering the normal of the level lines as in the Bregman process). This information is used in a second step that looks for an image that is the closest (relatively to l^2) to the observed data while it has the same relative order as the one obtained from the flow image. Interestingly enough, the imposition of the relative order is achieved through a Bregman distance. Note that our approach is a forward scale-space approach (as opposed to inverse as in [20]).

The remainder of this paper is as follows: some useful notations and definitions are given in section 2 while some theoretical results about differential inclusions are presented in section 3. Our approach is described in section 4. Finally we draw some conclusions in section 5.

2 Some Notations and Definitions

A discrete Markovian framework is considered [25]. An image is defined on a lattice referred to as \mathcal{V} with cardinality $|\mathcal{V}| = N$. We see any image as a vector living in \mathbb{R}^N . The value of an image \mathbf{u} at the site $i \in \mathcal{V}$ is referred to as u_i . We endow the lattice with a neighborhood system and we consider pairwise interactions between sites. We denote by (i, j) the interaction between two sites i and j that are neighbors of each other (relatively to the neighborhood system). The set of all of pairwise interactions is denoted by \mathcal{W} and we set $|\mathcal{W}| = M$.

In this paper we consider first order Discrete Total Variation (DTV) energies [3,6,11] with separable quadratic fidelity. In order to define first order DTVs, let us first follow [21] by noting that a discrete gradient, denoted by ∇ , is defined on any lattice endowed with a neighborhood system. This discrete gradient measures the variation for any two pixels in interaction and is thus a linear mapping $\nabla : \mathbb{R}^N \rightarrow \mathbb{R}^M$. In the remainder of this paper we should consider the matrix representation of ∇ . The adjoint operator of the discrete gradient corresponds to the divergence denoted by div and is defined as usual as

$$\langle \mathbf{u}, \text{div } \mathbf{w} \rangle_N = \langle -\nabla \mathbf{u}, \mathbf{w} \rangle_M.$$

We refer the reader to [21] for more details and properties of these linear operators. In this paper we define the Discrete Total Variation $J(\mathbf{u})$ for any image \mathbf{u} as the l^1 norm of the discrete gradient and is thus defined as

$$J(\mathbf{u}) = \|\nabla \mathbf{u}\|_{1,N} = \sum_{(i,j) \in \mathcal{W}} R_{i,j}(\mathbf{u}) = \sum_{(i,j) \in \mathcal{W}} |u_j - u_i|, \tag{1}$$

where each function $R_{i,j} : \mathbf{u} \in \mathbb{R}^N \mapsto |u_j - u_i| \in \mathbb{R}$ is associated with every single interaction (i, j) . Note that we have only considered non-weighted DTVs for the sake of clarity. Indeed, all results presented in this paper also holds for weighted versions of DTVs (i.e., each interacting term is weighted with a non-negative coefficient).

Finally, let us introduce the notion of subgradient and subdifferential. Let us consider $F : \mathbb{R}^N \rightarrow \bar{\mathbb{R}}$, a proper convex lower semi-continuous function. The *subdifferential* of F at the point \mathbf{x} corresponds to a set, denoted by $\partial F(\mathbf{x})$, defined as the following:

$$\partial F(\mathbf{x}) = \{\mathbf{s} \in \mathbb{R}^N \mid \forall \mathbf{y} \in \mathbb{R}^N, \langle \mathbf{y} - \mathbf{x}, \mathbf{s} \rangle + F(\mathbf{x}) \leq F(\mathbf{y})\}.$$

When F is differentiable at the point \mathbf{x} then $\partial F(\mathbf{x})$ is reduced to a singleton that contains the gradient of F evaluated at \mathbf{x} . Any element of $\partial F(\mathbf{x})$ is called a *subgradient* of F at \mathbf{x} . Also, for all $p \in \{1, \dots, N\}$ and $\mathbf{x} \in \mathbb{R}^N$, $\partial_p F(\mathbf{x})$ denoted the subdifferential at \mathbf{x} of the partial function $F_{(x_k)_{k \neq p}} : u \in \mathbb{R} \mapsto F(x_1, \dots, x_{p-1}, u, x_{p+1}, \dots, x_N)$. We refer the reader to [15],[16] for a complete study of these notions.

3 Differential Inclusion

This section is devoted to the presentation of the differential inclusion that describes the trajectory of the first process, i.e., the DTV-flow. We first recall some theoretical results that are essential for the theoretical soundness of our approach as well as for the computational point of view.

We consider the following differential inclusion problem that formally writes as

$$\begin{cases} \frac{d\mathbf{u}}{dt}(t) \in -\partial J(\mathbf{u}(t)) \text{ on } (0, +\infty) \\ \mathbf{u}(0) = \mathbf{v} \end{cases}, \tag{2}$$

Note that the non-differentiability of J may lead to several possible solutions. Following [2], we should consider the slowest solution that essentially consists in considering the subgradient of J at the current point with minimal Euclidean norm instead of the full subdifferential. Such a point of view not only yields a solution of the differential inclusion but also reduces it to a differential equation. We now list some fundamental properties that are proved for instance in [2].

Proposition 1. *Let $F : \mathbb{R}^N \rightarrow \bar{\mathbb{R}}$ be a proper lower semicontinuous convex function. The differential inclusion (2) has a unique absolutely continuous solution $\mathbf{u}(\cdot) : [0, +\infty) \rightarrow \mathbb{R}$ that satisfies*

$$\frac{d\mathbf{u}}{dt}(t) = -m(\partial F(\mathbf{u}(t))) \text{ for all } t \geq 0, \tag{3}$$

where $m(\partial F(\mathbf{u}(t)))$ is the orthogonal projection of the origin onto $\partial F(\mathbf{u}(t))$. Besides, the following holds:

- (i) $t \mapsto \|\mathbf{u}'(t)\|$ is nonincreasing.
- (ii) $t \mapsto \mathbf{u}(t)$ converges toward a minimizer of F .

In this paper we are interested in the trajectory generated when F corresponds to a discrete Total Variation given by (1) and where the initial condition is set to the observed data. This process generates a trajectory which simplifies the image and eventually converges toward a constant image. This yields a Discrete Total Variation flow that has been used in image processing (see for instance [14]).

Let us just note that the whole trajectory using DTV can be exactly computed using a network flow approach [21]. Such an approach relies on the fact that the trajectory $t \mapsto \mathbf{u}(t)$ is piecewise affine (one way to show this piecewise affine property is to reformulate DTV such that it fits the form of equation (3.4.7) of [15, p. 380]) We refer the reader to [9] for more details on the computations of such a trajectory.

4 A Discrete Total Variation Scale Space Approach

This section describes our approach. First, we show how to preserve the relative order of an image as a variational formulation using Bregman distance. Then, we describe how the latter is coupled with the differential inclusion to yield a contrast preserving denoising approach. Some numerical results are eventually presented.

4.1 Relative Order and Bregman Distances

The goal of this subsection is to characterize the set of all images which have the same *relative order* with respect to a given image using generalized Bregman distances.

Two images \mathbf{u} and \mathbf{v} are said to have the same relative order if and only if the following assertion holds: For all interactions $(i, j) \in \mathcal{W}$ we have

$$(\text{If } v_i < (>)v_j \text{ then } u_i \leq (\geq)u_j) \text{ or } (\text{If } v_i = v_j \text{ then } u_i = u_j) . \tag{4}$$

Following [20], we introduce the generalized Bregman distance which is a multi-valued version of the standard Bregman distance (which is originally defined for differentiable convex functions) based on the notion of subgradients. More precisely, the generalized Bregman distance $D_G(\mathbf{u}, \mathbf{v})$ between \mathbf{u} and \mathbf{v} , relatively to the lower semi-continuous function $G : \mathbb{R}^N \rightarrow \overline{\mathbb{R}}$, is the set of all quantities defined as the following:

$$D_G^{\mathbf{p}(\mathbf{v})}(\mathbf{u}, \mathbf{v}) = G(\mathbf{u}) - G(\mathbf{v}) - \langle \mathbf{p}(\mathbf{v}) | \mathbf{u} - \mathbf{v} \rangle_N, \tag{5}$$

where $\mathbf{p}(\mathbf{v}) \in \partial G(\mathbf{v})$.

Our goal is to describe the relative order set with the help of generalized Bregman distance for DTV, i.e., with $G = J$. More precisely, we will show that by choosing an adequate particular subdifferential $\mathbf{p}(\mathbf{v})$ (among all possible ones in the set $\partial J(\mathbf{v})$), then any image \mathbf{u} which has the same *relative order* with respect to \mathbf{v} satisfies $D_J^{\mathbf{p}(\mathbf{v})}(\mathbf{u}, \mathbf{v}) = 0$. To determine this subgradient, we use the combinatorial structure of DTV. We show that it is enough to consider the subgradients of the functions $R_{i,j}$ associated to each interaction (i, j) . Besides, we are able to express this set with generalized Bregman distance associated with each function $R_{i,j}$.

Let us detail this step. Consider any function $R_{i,j} : \mathbb{R}^N \rightarrow \mathbb{R}$ associated to the interaction (i, j) and its associated generalized Bregman distance. In order to define the latter (following the definition given by (5)), we need to pick an element of the subgradient of $R_{i,j}$ evaluated at the point \mathbf{v} . We consider the subgradient of minimal Euclidean norm living in $\partial R_{i,j}(\mathbf{v})$ that is referred to as $m(\partial R_{i,j}(\mathbf{v}))$. The choice of picking the one with minimal Euclidean norm is essential for our approach and will be justified later. The following proposition which relates DTV generalized Bregman distances and relative order is essential for our approach.

Proposition 2. *Let \mathbf{v} be an image. An image \mathbf{u} has the same relative order as \mathbf{v} if and only if \mathbf{u} verifies the following equalities:*

$$\forall (i, j) \in \mathcal{W}, \quad D_{R_{i,j}}^{m(\partial R_{i,j}(\mathbf{v}))}(\mathbf{u}, \mathbf{v}) = |u_j - u_i| + m_i(\partial R_{i,j}(\mathbf{v}))(u_j - u_i) = 0, \tag{6}$$

where $m_i(\partial R_{i,j}(\mathbf{v}))$ is the value of the image $m(\partial R_{i,j}(\mathbf{v}))$ at the site i , i.e., $m_i(\partial R_{i,j}(\mathbf{v})) = (m(\partial R_{i,j}(\mathbf{v})))_i$.

The proof of this proposition is given in Appendix A. Let us note that the proposition may fail to hold for if some particular choice of subgradients if it is not the one of minimal Euclidean norm. Indeed, equality (6) rewrites for any $\mathbf{p}(\mathbf{v}) \in \partial(R_{i,j}(\mathbf{v}))$ and $\mathbf{u} \in \mathbb{R}^N$ as the following (see Appendix A):

$$D_{R_{i,j}}^{\mathbf{p}(\mathbf{v})}(\mathbf{u}, \mathbf{v}) = |u_j - u_i| + p_i(\mathbf{v})(u_j - u_i) = 0, \tag{7}$$

where $p_i(\mathbf{v}) = (\mathbf{p}(\mathbf{v}))_i$ (i.e., the value of the image $\mathbf{p}(\mathbf{v})$ at the site i). Now let us consider only interactions for which we have $v_i = v_j$. For those interactions we have that $p_i(\mathbf{v}) \in [-1, 1]$. We shall see that equality (7) is not necessarily equivalent to the relative order property, depending on the element picked in the subgradient:

1. If $p_i(\mathbf{v}) = 1$, then, according to (7) we have : $D_{R_{i,j}}^{\mathbf{p}(\mathbf{v})}(\mathbf{u}, \mathbf{v}) = 0 \Leftrightarrow u_i \geq u_j$. Thus, there are some images satisfying equality constraint (7) which do not satisfy the relative order of image \mathbf{v} for the interaction (i, j) .

2. If $p_i(\mathbf{v}) = -1$, then, according to (7) we have : $D_{R_{i,j}}^{\mathbf{p}(\mathbf{v})}(\mathbf{u}, \mathbf{v}) = 0 \Leftrightarrow u_i \leq u_j$. As for the previous case, the equivalence may not hold.
3. If $p_i(\mathbf{v}) \in]-1, 1[$, then according to (7) we have : $D_{R_{i,j}}^{\mathbf{p}(\mathbf{v})}(\mathbf{u}, \mathbf{v}) = 0 \Leftrightarrow u_i = u_j$. This mean that all images satisfying equality constraint (7) follows the relative order of the image \mathbf{v} for the interaction (i, j) .

For cases where $v_i \neq v_j$ the subdifferential is reduced to one element and it is easy to check that the equivalence holds. Therefore, instead of picking the subgradient with minimal norm we could have chosen any one for which the i^{th} component is different from 1 or -1 when $v_i = v_j$ (and the pick the only possible subgradient for the other cases). We choose the one with minimal norm not only because it makes the equivalence to hold but also because it is essential for the convergence and thus soundness of our approach.

4.2 A Coupled Scale-Space Approach

We are now ready for describing our approach.

First we consider the unique slow solution of the differential inclusion (2) with discrete Total Variations, i.e., $F = J$ with the observed image \mathbf{v} as the initial condition. Let us note this solution as $t \mapsto u(t)$, and we consider the sequence $(\mathbf{u}(t))_{t \geq 0}$. As the solution corresponds to a Discrete Total Variation flow, we have that a scale space is generated such that the original image is more and more simplified as time evolves. Although one can only use DTV-flow for denoising such as in [13] for instance, it also suffers a loss of contrast [17] as for denoising by minimizing Total Variation with quadratic data fidelity term [23]. Thus, we wish to recover the contrast while keeping the important features contained in the flow.

For this purpose, we introduce a second sequence $(\tilde{\mathbf{u}}(t))_{t \geq 0}$ that are obtained from $(\mathbf{u}(t))_{t \geq 0}$. The most important features we wish to preserve in this paper is the shape of the level so that no geometric information will be introduced. Besides, we also wish to maintain the relative order of gray-level value between two adjacent pixels. Such constraints correspond to maintaining a *relative order*. In order to recover the contrast we wish to search for the image that is the closest (in the sense of the Euclidean norm) to the observed one while satisfying the relative order.

More precisely, we generate the sequence $(\tilde{\mathbf{u}}(t))_{t \geq 0}$ from $(\mathbf{u}(t))_{t \geq 0}$ such that for all $t \in \mathbb{R}^+$ the image $\tilde{\mathbf{u}}(t)$ is the unique solution of the following quadratic convex constrained problem

$$\tilde{\mathbf{u}}(t) := \arg \min_{\mathbf{g} \in S(t)} \left\{ \frac{1}{2} \|\mathbf{g} - \mathbf{v}\|_2^2 \right\}, \tag{8}$$

which corresponds to the projection of observed image \mathbf{v} into closed convex set $S(t)$. This set corresponds to images whose relative orders the same as $\mathbf{u}(t)$. It is formally defined as the following:

1. Set $\mathbf{u}(0) = \tilde{\mathbf{u}}(0) = \mathbf{v}$
2. Solve the differential inclusion problem

$$\begin{cases} \mathbf{u}'(t) \in -\partial J(\mathbf{u}(t)) & \text{on } (0, +\infty) \\ \mathbf{u}(0) = \mathbf{v} \end{cases},$$

3. For all $(i, j) \in \mathcal{W}$, compute $m_i(\partial R_{i,j})$ as below

$$m_i(\partial R_{i,j}(\mathbf{u}(t))) = \begin{cases} -1 & \text{if } \mathbf{u}(t)_i > \mathbf{u}(t)_j, \\ 0 & \text{if } \mathbf{u}(t)_i = \mathbf{u}(t)_j, \\ 1 & \text{if } \mathbf{u}(t)_i < \mathbf{u}(t)_j. \end{cases}$$

4. Solve

$$\tilde{\mathbf{u}}(t) := \arg \min_{\mathbf{g} \in S(t)} \left\{ \frac{1}{2} \|\mathbf{g} - \mathbf{v}\|_2^2 \right\}$$

Fig. 1. Our procedure

$$\begin{aligned} S(t) &= \bigcap_{(i,j) \in \mathcal{W}} \left\{ \mathbf{g} \in \mathbb{R}^N \mid D_{R_{i,j}}^{m(\partial R_{i,j}(\mathbf{u}(t)))}(\mathbf{g}, \mathbf{u}(t)) = 0 \right\} \\ &= \bigcap_{(i,j) \in \mathcal{W}} \left\{ \mathbf{g} \in \mathbb{R}^N \mid |g_j - g_i| + m_i(\partial R_{i,j}(\mathbf{u}(t)))(g_j - g_i) = 0 \right\}, \end{aligned} \quad (9)$$

where $m_i(\partial R_{i,j}(\mathbf{u}(t)))$ is given by

$$m_i(\partial R_{i,j}(\mathbf{u}(t))) = \begin{cases} -1 & \text{if } \mathbf{u}(t)_i > \mathbf{u}(t)_j, \\ 0 & \text{if } \mathbf{u}(t)_i = \mathbf{u}(t)_j, \\ 1 & \text{if } \mathbf{u}(t)_i < \mathbf{u}(t)_j. \end{cases}$$

Our approach is summarized in Figure 1. We can further give a result of convergence for this algorithm which is summarized in the following proposition.

Proposition 3. *Assume that the observed image \mathbf{v} has zero mean, then the sequence of images $(\tilde{\mathbf{u}}(t))_{t>0}$ generated by previous theoretical algorithm converges toward the constant image $\mathbf{0}$.*

We only give the fundamental elements of the proof. According to theorem 1 of [2, p.147], theorem 1 of [2, p.149] and theorem 2 of [2, p.160], we prove that $\lim_n m(\partial R_{i,j}(\mathbf{u}(t))) = \mathbf{0}$ and also that the sequence $(\tilde{\mathbf{u}}(t))_{t>0}$ is bounded. According to theorem 4.18 [22, p.120] and proposition 4.9 [22, p.120] we conclude that $\mathbf{0}$ is unique cluster point of sequence $(\tilde{\mathbf{u}}(t))_{t>0}$.

Proposition 3 yields a natural stopping rule. Assuming the variance σ^2 of the noise that corrupts the image is known, one natural stopping criteria is to stop the process as soon as the residual satisfies $\|\tilde{\mathbf{u}}(t) - \mathbf{v}\|_{2,N}^2 \geq N\sigma^2$. Note that if $\|\mathbf{v}\|_{2,N}^2 > N\sigma^2$ then the stopping criteria is never met and thus the denoised image is the null image (this behavior is similar to the existence of Lagrange

multiplier for Total Variation minimization with quadratic constraint [7]). Also, although the function $t \mapsto \mathbf{u}(t)$ is a continuous function, $t \mapsto \tilde{\mathbf{u}}(t)$ might not be (actually it is not for practical interesting cases).

4.3 Numerical Results

From a practical point of view we take benefit from the fact that the solution of the differential inclusion with Discrete Total Variation given in (2) is piecewise affine. It is thus enough to detect specific times where the subgradient of J with minimal norm is changing to be able to generate the trajectory. This can be easily computed using a network flow approach (see [9]). On a time interval on which the solution of the differential inclusion is affine, its subgradient of minimal Euclidean norm is constant and thus the convex constraint set $S(\cdot)$ is constant of this interval, and thus the solution $\tilde{\mathbf{u}}(\cdot)$ is constant on this interval. It can further be shown that there is a finite number of specific time for which the subgradient of minimal Euclidean norm is changing. Thus, the implementation detects these specific time, compute the solution $\tilde{\mathbf{u}}$ at this time. It iterates until the stopping criteria is met (or the subgradient of minimal Euclidean norm is zero).

In this experiment, we consider Discrete Total Variations that involves the 8-connectivity. The weights for the 4 nearest neighbors are set to 1, while the weights for the diagonal ones are set to $1/\sqrt{2}$.

Figure 2 depicts an original image and its noisy version corrupted by an independent Gaussian additive noise of zero mean and variance $\sigma^2 = 12^2$. The result using our approach is depicted in Figure 3 along with its residual. One can see that the residual seems to contain very little geometric information. The mean square error (MSE) is 48.6191.

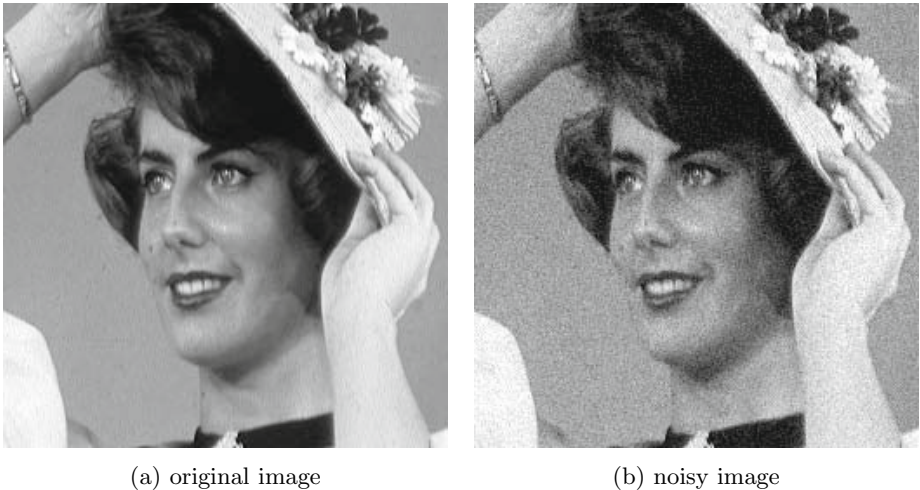


Fig. 2. Original image is depicted in (a) while its corrupted version by an additive Gaussian noise (zero mean) of variance $\sigma^2 = 12^2$ is depicted in (b)

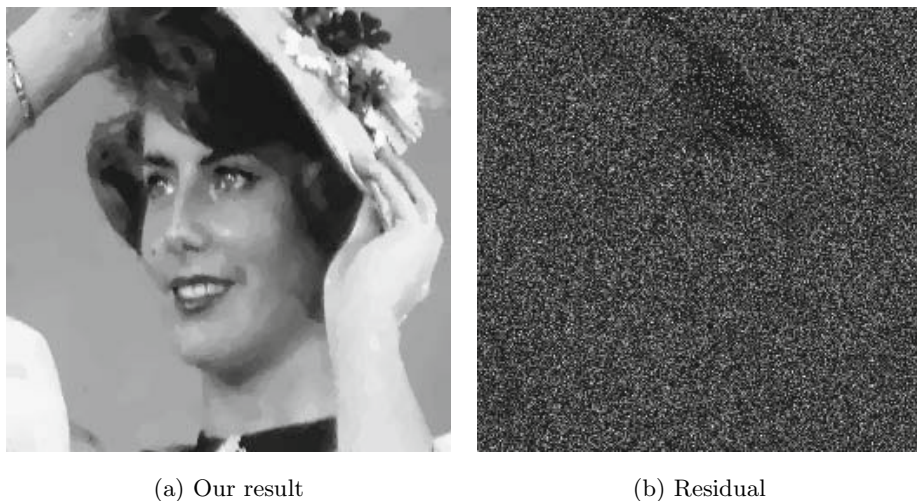


Fig. 3. Result using our approach: The denoised image and its difference with the noisy image are respectively depicted in (a) and (b)



Fig. 4. Total Variation result: The minimizer and its difference with the noisy image are respectively depicted in (a) and (b)

For comparison purposes we also present in Figure 4 the result obtained using a standard variational model involving Discrete Total Variation with a quadratic constraint ; we refer the reader to [7,23] more details. The MSE for this approach is 49.26. Although the MSE for both approaches are essentially the same it is clearly seen that the residual is much better for approach as it contains much less geometrical detail.

5 Conclusion

This paper has described an approach to restore images while preserving the contrast. It consists on coupling the solution a simplification process, here a DTV-flow, with another process that recovers the contrast. We have shown the well-soundness and the convergence of our approach.

Further extension consists in replacing the Total Variation flow by more elaborated flows. Also, although our approach is a forward scale-space it would be interesting to better understand the theoretical links, if any, with the *inverse* scale space of Osher et al. [20].

Acknowledgement. Research of Jérôme Darbon has been supported by ONR N000140710810. Jérôme Darbon wishes to thank Gary Hewer (NAVAIR Weapons Division) for fruitful discussions and his support.

References

1. Aubert, G., Kornprobst, P.: *Mathematical Problems in Image Processing*. Springer, Heidelberg (2002)
2. Aubin, J.P., Cellina, A.: *Differential Inclusions*. Springer, Heidelberg (1984)
3. Bouman, C., Sauer, K.: A generalized gaussian image model for edge-preserving map estimation. *IEEE Transactions on Signal Processing* 2(3), 296–310 (1993)
4. Burger, M., Gilboa, G., Osher, S., Xu, J.: Nonlinear inverse scale space methods. *Communications in Mathematical Sciences* 5(1), 175–208 (2006)
5. Chalmond, B.: *Modeling and Inverse Problems in Image Analysis*. Springer, Heidelberg (2003)
6. Chambolle, A., Darbon, J.: On total variation minimization and surface evolution using parametric maximum flows. *International Journal of Computer Vision* 84(3), 288–307 (2009)
7. Chambolle, A., Lions, P.L.: Image recovery via total variation minimization and related problems. *Num. Math.* (76), 167–188 (1997)
8. Charbonnier, P., Blanc-Féraud, L., Barlaud, M., Aubert, G.: Deterministic edge-preserving regularization in computed imaging. *IEEE Transactions on Image Processing* 5(2), 298–311 (1997)
9. Darbon, J.: In preparation. Tech. rep. (2010)
10. Darbon, J., Ciril, I., Marquina, A., Chan, T., Osher, S.: A note on the bregmanized total variation and dual forms. In: 16th IEEE International Conference on Image Processing (ICIP), pp. 2965–2968 (November 2009)
11. Darbon, J., Sigelle, M.: Image restoration with discrete constrained Total Variation part I: Fast and exact optimization. *Journal of Mathematical Imaging and Vision* 26(3), 261–276 (2006)
12. Geman, D., Reynolds, G.: Constrained restoration and the recovery of discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14(3), 367–383 (1992)
13. Gilboa, G., Darbon, J., Osher, S., Chan, T.F.: Nonlocal convex functionals for image regularization. Tech. Rep. CAM-06-57, UCLA (October 2006)

14. Gilboa, G., Sochen, N., Zeevi, Y.Y.: Forward-and-backward diffusion processes for adaptive image enhancement and denoising. *IEEE Transactions on Image Processing* 11(2), 689–703 (2002)
15. Hiriart-Urruty, J.B., Lemaréchal, C.: *Convex Analysis and Minimization Algorithms Part I*. Springer, Heidelberg (1996)
16. Hiriart-Urruty, J.B., Lemaréchal, C.: *Convex Analysis and Minimization Algorithms Part II*. Springer, Heidelberg (1996)
17. Meyer, Y.: *Oscillating patterns in image processing and nonlinear evolution equations*. University Lecture Series, vol. 22. American Mathematical Society, Providence (2001); the fifteenth Dean Jacqueline B. Lewis memorial lectures
18. Nikolova, M., Chan, R.H.: The equivalence of half-quadratic minimization and the gradient linearization iteration. *IEEE Transactions on Image Processing* 16(6), 1623–1627 (2007)
19. Nikolova, M., Ng, M.: Analysis of half-quadratic minimization methods for signal and image recovery. *SIAM Journal on Scientific Computing* 27(3), 937–9660 (2005)
20. Osher, S., Burger, M., Goldfarb, D., Xu, J., Yin, W.: An Iterative Regularization Method for Total Variation Based Image Restoration. *SIAM Journal on Multiscale Modeling and Application* 4, 460–489 (2005)
21. Rockafellar, R.T.: *Network Flows and Monotropic Optimization*. John Wiley & Sons, Chichester (1984)
22. Rockafellar, R.T., Wets, R.: *Variational Analysis*. Springer, Heidelberg (1998)
23. Rudin, L., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. *Physica D* 60, 259–268 (1992)
24. Strong, D., Chan, T.F.: Edge-Preserving and Scale-Dependent Properties of Total Variation Regularization. *Inverse Problems* 9, S165–S187 (2000)
25. Winkler, G.: *Image Analysis, Random Fields and Dynamic Monte Carlo Methods*. Applications of Mathematics. Springer, Heidelberg (2003)

A Appendix

Proof of proposition 2: Let $\mathbf{v} \in \mathbb{R}^N$, be an image and consider an interaction $(i, j) \in \mathcal{W}$. First, for all $\mathbf{p}(\mathbf{v}) \in \partial R_{i,j}(\mathbf{v})$ and $\mathbf{u} \in \mathbb{R}^N$, according to proposition 11.3 [22] we can rewrite $D_{R_{i,j}}^{\mathbf{p}(\mathbf{v})}(\mathbf{u}, \mathbf{v})$ defined by (5) as below :

$$D_{R_{i,j}}^{\mathbf{p}(\mathbf{v})}(\mathbf{u}, \mathbf{v}) = R_{i,j}(\mathbf{u}) + R_{i,j}^*(\mathbf{p}(\mathbf{v})) - \langle \mathbf{p}(\mathbf{v}) | \mathbf{u} \rangle_N \quad , \quad (10)$$

where $R_{i,j}(\mathbf{u}) = r(\mathbf{d}_{i,j}^t \mathbf{u})$, with $r : x \in \mathbb{R} \mapsto |x|$ and $\mathbf{d}_{i,j} \in \mathbb{R}^N$ defined such that: $(\mathbf{d}_{i,j})_i = -1$, $(\mathbf{d}_{i,j})_j = 1$ and for all $p \in \{1, \dots, N\} - \{i, j\}$ $(\mathbf{d}_{i,j})_p = 0$. Also, by invoking [22] theorem 11.23(b) we have :

$$R_{i,j}^*(\mathbf{p}(\mathbf{v})) = \inf_{w \in \mathbb{R}} \{ r^*(w) \mid -w \mathbf{d}_{i,j} = \mathbf{p}(\mathbf{v}) \} \quad , \quad (11)$$

where :

$$r^*(x) = \begin{cases} 0 & \text{if } |x| \leq 1 \\ +\infty & \text{otherwise} \end{cases} \quad . \quad (12)$$

As $\|\mathbf{p}(\mathbf{v})\|_\infty \leq 1$, and according to (11) and (12), the generalized Bregman distance (10) becomes:

$$D_{R_{i,j}}^{\mathbf{p}(\mathbf{v})}(\mathbf{u}, \mathbf{v}) = \inf_{w \in \mathbb{R}} \{ |u_j - u_i| + r^*(w) + w \langle \mathbf{d}_{i,j} | \mathbf{u} \rangle_N - w \mathbf{d}_{i,j} = \mathbf{p}(\mathbf{v}) \} \\ = |u_j - u_i| + p_i(\mathbf{v})(u_j - u_i) . \tag{13}$$

By a simple subdifferential calculus we have:

$$\partial_i R_{i,j}(\mathbf{v}) = \begin{cases} -1 & \text{if } v_i > v_j \\ [-1, 1] & \text{if } v_i = v_j \\ 1 & \text{if } v_i < v_j \end{cases} \Rightarrow m_i(\partial R_{i,j}(\mathbf{v})) = \begin{cases} -1 & \text{if } v_i > v_j \\ 0 & \text{if } v_i = v_j \\ 1 & \text{if } v_i < v_j \end{cases} \tag{14}$$

Therefore, taking $\mathbf{p}(\mathbf{v}) = m(\partial R_{i,j}(\mathbf{v}))$ in (13), and according to (14), we have the following three cases:

1. If $v_i > v_j$, we have the following equivalences:

$$D_{R_{i,j}}^{m(\partial R_{i,j}(\mathbf{v}))}(\mathbf{u}, \mathbf{v}) = |u_j - u_i| + (u_j - u_i) = 0 \Leftrightarrow u_i - u_j = |u_j - u_i| \Leftrightarrow u_i \geq u_j .$$

2. If $v_i < v_j$, we have the following equivalences:

$$D_{R_{i,j}}^{m(\partial R_{i,j}(\mathbf{v}))}(\mathbf{u}, \mathbf{v}) = |u_j - u_i| - (u_j - u_i) = 0 \Leftrightarrow u_j - u_i = |u_j - u_i| \Leftrightarrow u_j \geq u_i .$$

3. If $v_i = v_j$, we have the following equivalence:

$$D_{R_{i,j}}^{m(\partial R_{i,j}(\mathbf{v}))}(\mathbf{u}, \mathbf{v}) = |u_j - u_i| = 0 \Leftrightarrow u_i = u_j .$$

Therefore, the equivalence stated in proposition 2 is proved. □

Smale-Like Decomposition and Forman Theory for Discrete Scalar Fields

Lidija Čomić¹, Mohammed Mostefa Mesmoudi², and Leila De Floriani²

¹ Faculty of Engineering, University of Novi Sad, Serbia

² Department of Computer Science, University of Genova, Italy

Abstract. Forman theory, which is a discrete alternative for cell complexes to the well-known Morse theory, is currently finding several applications in areas where the data to be handled are discrete, such as image processing and computer graphics. Here, we show that a discrete scalar field f , defined on the vertices of a triangulated multidimensional domain Σ , and its gradient vector field $Grad f$ through the *Smale-like decomposition* of f [6], are both the restriction of a Forman function F and its gradient field $Grad F$ that extends f over all the simplexes of Σ . We present an algorithm that gives an explicit construction of such an extension. Hence, the scalar field f inherits the properties of Forman gradient vector fields and functions from field $Grad F$ and function F .

Keywords: Morse Theory, Forman Theory, Morse Decomposition.

1 Introduction

Morse theory is a powerful tool for understanding the topology and the geometry of a manifold M on which a C^2 -differentiable real-valued function f is defined. A Morse function f induces decompositions of M (called Morse complexes) into regions associated with critical points of f , based on the study of the behavior of the gradient vector field of f . In 1998, Forman introduced a new theory for cell complexes, that is a discrete equivalent to Morse theory [8]. He proved that almost all the main results from Morse theory are valid for discrete functions.

We describe a discrete decomposition for a triangulated n -dimensional domain Σ with manifold carrier, associated with a scalar field f and originally proposed in [6], called a *Smale-like decomposition*. This decomposition simulates the Morse complexes of f in the discrete case, and defines a discrete gradient field $Grad f$, which represents the topological structure of the field. We have used such decomposition in combination with simplification operations to build a multi-scale morphological representation of scalar fields [2, 5].

We construct an extended form of discrete gradient field $Grad f$ defined by a Smale-like decomposition, called an *extended discrete gradient field* $EGrad f$. The extended form always points in the direction in which function f is descending, and thus it agrees with the (negative) flow induced by the scalar field f . We show that this field is a Forman gradient vector field V_F of a Forman function F , whose restriction over the vertices of Σ coincides with the initial scalar field f .

We give the explicit formulation of a Forman function F that satisfies the above property.

As a consequence, we have that it is possible to use all the machinery of Forman theory without actually working with the entire Forman function F , or even with the related Forman gradient vector field $V_F = EGrad f$; the vector sub-field $Grad f$ is sufficient to study the behavior of scalar field f . Specifically, $EGrad f$ (and thus F) can be simplified by applying a cancellation operator, which eliminates critical cells of $EGrad f$ in pairs. This simplification operator will form a basis of a hierarchical representation of the morphology of the initial scalar field f , and of the simplicial complex Σ . We intend to build such a hierarchical model, based on a cancellation operator, in the near future.

The remainder of this paper is organized as follows. In Sections 2 and 3, we summarize some results related to Morse and Forman theory, respectively. In Section 4, we review related work in this area. In Section 5, we recall the algorithm that computes the Smale-like decomposition. In Section 6, we present the algorithmic construction of a discrete vector field $EGrad f$, which extends the discrete gradient field $Grad f$ induced by the Smale-like decomposition to a Forman gradient vector field. In Section 7, we define a Forman function F , whose Forman gradient vector field V_F coincides with $EGrad f$. In Section 8, we draw some concluding remarks and discuss our current and future work.

2 Morse Theory and Morse Complexes

Morse theory captures the relationship between the topology of a manifold M and the critical points of a scalar function f defined on the manifold [13, 14].

Let f be a C^2 real-valued function defined over a closed compact n -manifold M . A point p is a *critical point* of f if and only if the gradient $\nabla f = (\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n})$ (in some local coordinate system around p) of f vanishes at p . Function f is a *Morse function* if all its critical points are non-degenerate. The number i of negative eigenvalues of the Hessian matrix $Hess_p f$ is called the *index* of critical point p , and p is called an *i -saddle*. A 0-saddle (an n -saddle) is also called a *minimum* (a *maximum*). An *integral line* of f is a maximal path which is everywhere tangent to ∇f . Each integral line starts and ends at critical points of f , called its *origin* and its *destination*.

Integral lines that converge to (originate at) a critical point p of index i form an i -cell ($(n-i)$ -cell), called a *descending* (*ascending*) *cell* of p . The descending and ascending cells decompose M into *descending* and *ascending Morse complexes*, denoted as Γ_d and Γ_a , respectively. A Morse function f is called a *Morse-Smale function* if each non-empty intersection of a descending and an ascending cell is transversal. These intersections define a *Morse-Smale complex*.

3 Forman Theory

Forman theory is a discrete counterpart of Morse theory, and its main purpose is to transpose the results of Morse theory from a smooth to a combinatorial

setting. A function F , defined on all simplexes (and not only on vertices) of a finite simplicial complex Σ , is called a *Forman function* if for any p -simplex σ , all the $(p - 1)$ -simplexes in the boundary of σ have a lower F value than σ , and all the $(p + 1)$ -simplexes in the coboundary of σ have a higher F value than σ , with at most one exception. A simplex is *critical* if there is no exception to this rule. More formally, a function $F : \Sigma \rightarrow \mathbb{R}$ is a *Forman function* if for every p -simplex σ , both the following conditions are satisfied

$$(1) \#\{\tau^{(p+1)} > \sigma : F(\tau) \leq F(\sigma)\} \leq 1, \quad (2) \#\{\nu^{(p-1)} < \sigma : F(\nu) \geq F(\sigma)\} \leq 1.$$

These inequalities cannot be equalities at the same time. A p -simplex $\sigma \in \Sigma$ is a *critical simplex* of index p if both the following conditions are satisfied

$$(1) \#\{\tau^{(p+1)} > \sigma : F(\tau) \leq F(\sigma)\} = 0, \quad (2) \#\{\nu^{(p-1)} < \sigma : F(\nu) \geq F(\sigma)\} = 0.$$

The absolute minimum of F on a a triangulation of a closed manifold occurs at a vertex; the absolute maximum occurs at a maximal dimensional simplex [8].

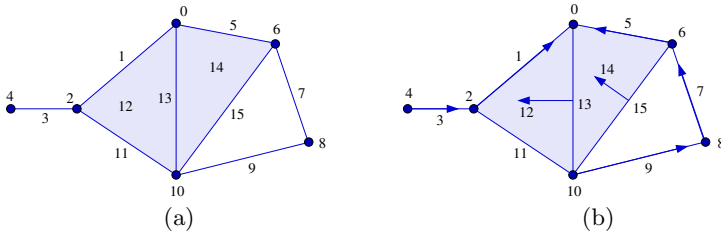


Fig. 1. (a) Forman function F , and (b) the corresponding discrete gradient vector field V_F , on a 2D simplicial complex. Each simplex σ is labelled by the value of F at σ .

In the example in Figure 1(a), a Forman function F defined on a 2D simplicial complex is illustrated. Each simplex is labelled by its function value. Vertex labelled 0 and edge labelled 11 are critical simplexes of F .

Forman theory can be introduced starting from a notion of a discrete vector field, which can be imagined as a collection of arrows, connecting a p -simplex of Σ to an incident $(p + 1)$ -simplex, such that each simplex is a head or a tail of at most one arrow. A simplex is critical if it is neither the head nor the tail of any arrow. A discrete vector field V is a Forman gradient vector field if there are no closed V -paths in V . More formally, a *discrete vector field* V on a simplicial complex Σ is a collection of pairs (σ, τ) , such that

- (1) σ is a p -simplex, and τ is a $(p + 1)$ -simplex of Σ ,
- (2) σ is a face of τ ($\sigma < \tau$), and
- (3) each simplex of Σ is in at most one pair of V .

A V -path is a sequence $\sigma_0, \tau_0, \sigma_1, \tau_1, \dots, \sigma_{r+1}$ of p -simplexes σ_i and $(p + 1)$ -simplexes τ_j , $i = 0, \dots, r + 1$, $j = 0, \dots, r$, such that $(\sigma_i, \tau_i) \in V$, $\tau_i > \sigma_{i+1}$, and $\sigma_i \neq \sigma_{i+1}$. A sequence $\sigma_0, \tau_0, \sigma_1, \tau_1, \dots, \sigma_{r+1}$, $r > 0$, is a *closed path* if it is a

V -path, and $\sigma_{\tau+1} = \sigma_0$. A discrete vector field V is called a *discrete (Forman) gradient vector field* if and only if there are no closed V -paths in V . A *critical simplex* of V of index p is a p -simplex σ which does not appear in any pair of V . In other words, a simplex σ is critical if $V(\sigma) = \emptyset$, and $\sigma \notin \text{Im}V$. A gradient vector field V is a subgraph (a forest) of the (non-oriented) Hasse diagram of Σ .

There is a correspondence between Forman functions and Forman gradient vector fields. For each Forman function F , a Forman gradient vector field V_F can be constructed, by drawing an arrow from a p -simplex σ to a $(p+1)$ -simplex τ (adding a pair (σ, τ) to V_F) if $\tau > \sigma$ and $F(\tau) \leq F(\sigma)$. Conversely, for each Forman gradient vector field V there exists a (non-unique) Forman function F such that the gradient field V_F of F is V [7]. The example in Figure 1(b) shows a Forman gradient vector field V_F of the Forman function F in Figure 1(a).

4 Related Work

There have been several proposals which construct a Forman gradient vector field, or a Forman function, on a simplicial complex Σ . Computation of a Forman gradient vector field V with minimal number of critical cells on a 2D simplicial complex Σ with a manifold domain has been discussed by Lewiner et al. [12], with the objective to compute the homology of Σ . It is not possible to analyze a specific scalar field f given on vertices of Σ using this approach. Forman theory has been used to build approximations of a Morse-Smale complex by Cazals et al. [1], with the objective to segment the surface of a molecule, using a discrete Connolly function f computed on each vertex of the surface.

In [11], King et al. consider the problem of computing a Forman gradient vector field V and the corresponding Forman function F , which extends a scalar field f given on the vertices of a triangulated surface. Simplification of V by cancellation of critical simplexes is included in the algorithm. Forman function F is arbitrarily close to the maximum of f over vertices of σ . Forman function which we construct here can be easily converted into another Forman function G , satisfying the same condition. As opposed to the gradient field $EGrad f$ which we construct, gradient field V constructed in [11] does not always point in the direction in which scalar field f is decreasing. The approach in [11] is extended to arbitrary dimensions by Jerše and Mramor Kosta in [10], and it is used to define a Forman gradient vector field V on a regular cell complex Γ with a manifold carrier $\Delta\Gamma$. Descending regions related to critical cells of V are defined. It is shown that after a finite number of subdivisions, all descending regions are topological disks. Ascending regions are defined using the dual complex.

In [9], Gyulassy et al. use Forman theory and a divide-and-conquer technique to compute an approximation of the Morse-Smale complex of a scalar field f defined on the vertices of a regular cell complex Γ with manifold carrier. It is not guaranteed that the constructed discrete gradient vector field V , if applied to a triangulated domain, points in the direction in which the scalar field f is decreasing. Forman function F corresponding to field V is not constructed.

In [3], Cousty et al. demonstrate a link between collapse operator for maps and watersheds on pseudomanifolds. It is assumed that a function F (which is not a Forman function) is defined on all simplexes of a pseudomanifold. A collapse is used on the level sets of F to obtain watershed of F .

5 The Smale-Like Discrete Decomposition

In this Section, we describe a dimension independent region-based method, introduced in [4], for extracting an approximation of a descending Morse complex related to maxima, called a *Smale-like decomposition*, starting from a scalar field f defined on the vertices of a triangulated manifold domain Σ .

The process of region-growing is based on the elevation values at the vertices of Σ , and the connectivity of Σ . It is assumed that $f(p) \neq f(q)$ if $p \neq q$ (p and q are vertices of Σ). This condition, which is assumed in other methods that produce a segmentation of Σ using Forman theory [9-11], ensures, over regular complexes, the uniqueness of the decomposition, and can be achieved by a small perturbation of scalar field f . The decomposition obtained will depend on the way the input data is perturbed. The algorithm extracts descending regions of a discrete Morse complex, related to maxima of f , and can be modified in an obvious way to obtain ascending regions related to minima. Overlay of ascending and descending regions defines an approximation of a discrete Morse-Smale complex.

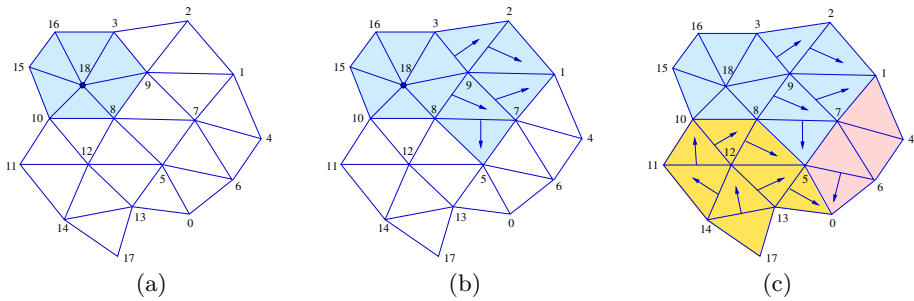


Fig. 2. (a) Initialization, and (b) the complete region associated with point p at elevation 18. (c) The decomposition of a two-dimensional triangulated domain D . Points are labelled by their function value. Arrows indicate the region growing process.

In the preprocessing step, the vertices of Σ are sorted in descending order of their elevation, and are processed in this order. A current set K is kept, which is initialized to be equal to Σ . A descending region $C(p)$, associated with global maximum p , is initialized with all the simplexes in $\overline{St}(p)$ (the closure of the star of p in K), and the boundary $\partial C(p)$ of C is equal to the link $Lk(p)$ in K . For example, in Figure 2(a), each vertex p of a 2D triangulated surface is labelled by

the value of a scalar field f at p . A global maximum of f is achieved at a point labelled 18, and the descending region $C(18)$ is initialized with all the simplexes in the closed star of vertex 18.

After the initialization step, the region growing step of the algorithm is performed. For each top $((n - 1)$ -dimensional) simplex $\gamma \in \partial C(p)$, which is incident to another simplex (cone) $q * \gamma \in K - C(p)$, if $f(q)$ is less than $f(r)$, for all vertices r of γ , then $C(p)$ is extended to $C(p) \cup \overline{q * \gamma}$, and the boundary $\partial C(p)$ of $C(p)$ is updated by replacing γ with all faces of the cone $q * \gamma$ that contain q . Region growing is illustrated in Figure 2(b). Region $C(p)$ is extended iteratively, until no more simplexes can be added to it while maintaining the above property. Then, the interior of $C(p)$ is deleted from K , and the process is repeated until there are no more n -simplexes in K . After all vertices of the example illustrated in Figure 2 are processed, decomposition given in Figure 2(c) is obtained. Figure 3 shows the results produced by the Smale-like decomposition algorithm on a 2D terrain data set, representing Mount Marcy (courtesy of USGS).

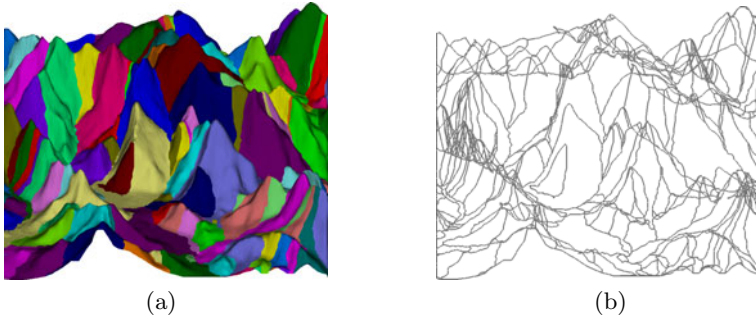


Fig. 3. (a) Perspective view of the 119 unstable components, produced by the Smale-like decomposition algorithm, applied to Mount Marcy with 69718 triangles. (b) Wire-frame model.

6 Extended Discrete Gradient Field

A discrete gradient field $Grad f$ can be defined using the decomposition algorithm described in the previous Section. It associates with each $(n - 1)$ -dimensional simplex γ , which has been used in the extension process, the cone $q * \gamma$ added to $C(p)$, i.e., $Grad f(\gamma) = q * \gamma$. Here, we extend the field $Grad f$ into a discrete gradient vector field $EGrad f$, defined on all simplexes of Σ , which we call *extended discrete gradient field*.

Intuitively, for each pair $(\gamma, q * \gamma)$, such that $Grad f(\gamma) = q * \gamma$, $EGrad f$ can be defined over all i -faces σ^i of $(n - 1)$ -simplex γ , $0 \leq i \leq n - 2$, by associating σ^i with cone $q * \sigma^i$, i.e., by adding the pair $(\sigma^i, q * \sigma^i)$ to $EGrad f$ (by setting $EGrad f(\sigma^i) := q * \sigma^i$). Geometrically, this extension consists of emanating vectors from all faces σ^i of γ towards vertex q . This is compatible with Smale-like decomposition process since $f(q) < f(r)$ for all vertices r of γ . Note that,

according to this construction, all faces of γ are not critical since they are tails of vectors. If any of such $(n - 1)$ -dimensional cones $q * \sigma^{n-2}$ is used later to expand $C(p)$ in the Smale-like decomposition process, the pair $(\sigma^{n-2}, q * \sigma^{n-2})$ has to be removed from $EGrad f$, and the $(n - 1)$ -dimensional cone $q * \sigma^{n-2}$ is processed in the same way as $(n - 1)$ -simplex γ above. In this way we avoid having a simplex σ such that there is an arrow starting at, and an arrow ending at σ . We now give a formal description of the step-by-step algorithm which defines a discrete field $EGrad f$. Steps (1), (2), and (3) define $EGrad f$ on $(n - 1)$ -dimensional, i -dimensional (for $0 \leq i \leq (n - 3)$), and $(n - 2)$ -dimensional simplexes of Σ , respectively.

Definition 1. *The extended discrete gradient field $EGrad f$ of a scalar function f (and of the corresponding discrete gradient field $Grad f$) defined on the vertices of an n -dimensional triangulated domain Σ is given by*

- (1) *if γ is an $(n - 1)$ -simplex such that $Grad f(\gamma) = q * \gamma$ (γ is expanding a component $C(p)$ to include the cone $q * \gamma$) then $EGrad f(\gamma) := q * \gamma$.*
- (2) *For all i -simplexes $\sigma^i < \gamma$, where $EGrad f(\gamma) = q * \gamma$ for some q , and $i = 0, \dots, n - 3$, $EGrad f(\sigma^i) := q * \sigma^i$ if $EGrad f(\sigma^i)$ has not been defined before when another $(n - 1)$ -simplex γ incident to σ^i was considered. Otherwise, σ^i is skipped (since it has already an attached value by $EGrad f$).*
- (3) *For all $(n - 2)$ -dimensional simplexes $\sigma^{n-2} < \gamma$, where $EGrad f(\gamma) = q * \gamma$ for some q , we distinguish two cases:*
 - (a) *If the $(n - 1)$ -dimensional cone $q * \sigma^{n-2}$ does not participate in the expansion process of $C(p)$, and if it is not already paired with some $(n - 2)$ -simplex in $EGrad f$ (if it is not in the image of $EGrad f$), then we set $EGrad f(\sigma^{n-2}) := q * \sigma^{n-2}$.*
 - (b) *Otherwise, we set (temporarily) $EGrad f(\sigma^{n-2}) := \emptyset$. In this case, cone $q * \sigma^{n-2}$ represents a new expanding $(n - 1)$ -simplex of $C(p)$. Return to steps (1), (2) and (3) to define $EGrad f$ on cone $q * \sigma^{n-2}$ (i.e., to set $EGrad f(q * \sigma^{n-2}) := Grad f(q * \sigma^{n-2})$), on i -faces of cone $q * \sigma^{n-2}$, $0 \leq i \leq n - 3$, and on $(n - 2)$ -faces of cone $q * \sigma^{n-2}$, respectively.*

For each simplex σ in the open star of a vertex p which starts a new component, such that σ does not belong to the boundary or interior of another component $C(t)$, σ is a critical simplex, and $EGrad f(\sigma) := \emptyset$. We note here that our extended field $EGrad f$ depends on the order in which top simplexes are processed. Such order dependence is common to other approaches which compute a discrete gradient vector field starting from a scalar function f [9, 11].

We present in Figure 4 the extended discrete gradient field $EGrad f$ induced by scalar field f , illustrated in Figure 2. Simplexes γ are edges and their faces σ^i are vertices (i.e., we have only $i = 0$). Edge $\gamma = [3; 9]$ expands the component $C(18)$, by adding triangle $[3; 9; 2]$ to $C(18)$, and $EGrad f([3; 9]) := [3; 9; 2]$ (step (1)). End points of edge $[3; 9]$ are the $(n - 2)$ -simplexes described above. Edge $[3; 2]$ does not participate in the expansion of the new component $C(18) := C(18) \cup [3; 9; 2]$. Thus, $EGrad f([3]) = [3; 2]$ (step (3a)). The other vertex $[9]$ of edge $[9; 2]$, with vertex $[2]$ forms an edge that expands the updated $C(18)$,

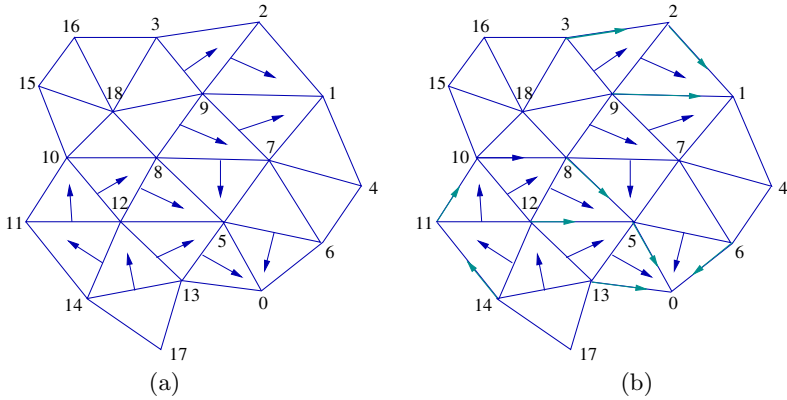


Fig. 4. (a) Function $Grad f$, defined by the Smale-like decomposition of Figure 2 (b) The corresponding extended discrete gradient field $EGrad f$.

so (temporarily) $EGrad f([9]) = \emptyset$ (step (3b)). Vertex [9] is visited again when triangle [9; 2; 1] is considered. Function $EGrad f$ associates vertex [9] with edge [9; 1], since edge [9; 1] does not expand $C(p)$ (step (3a)). Vertex [9] is revisited again when triangle [9; 1; 7] is considered. The process skips here vertex [9] since it has already a non-empty value by $EGrad f$ (step (2)). We have here one (global) minimum [0] and the entire stars $St(18)$ and $St(17)$, and a part of $St(7)$ as critical cells corresponding to local maxima 18 and 17.

Theorem 1. *The extended discrete gradient field $EGrad f$ defined by the algorithm in Definition 1 is a Forman gradient vector field.*

Proof. We need to show that (i) $EGrad f$ is a discrete vector field, and (ii) there are no closed $EGrad f$ -paths in Σ .

(i) From the construction of $EGrad f$ it is clear that $EGrad f$ pairs an i -simplex σ^i with an $(i + 1)$ -simplex (cone) $q * \sigma^i$, $0 \leq i \leq n - 1$, that simplex σ^i is a face of the cone $q * \sigma^i$, and that each simplex is a head or a tail of at most one arrow. Thus, $EGrad f$ is a discrete vector field.

(ii) Let us assume that the sequence $\sigma_0, \tau_0, \sigma_1, \tau_1, \dots, \sigma_r, \tau_r, \sigma_0$ of i -simplexes σ_k and $(i + 1)$ -simplexes τ_k , $k, l = 0, \dots, r > 0$, is a closed $EGrad f$ -path in Σ . Then $(\sigma_k, \tau_k) \in EGrad f$, (i.e., $EGrad f(\sigma_k) = \tau_k$), $\tau_k > \sigma_{k+1}$, and $\sigma_k \neq \sigma_{k+1}$. By the construction of $EGrad f$, and the condition that $f(p) \neq f(q)$ for any two vertices p and q of Σ , we have that $\tau_0 = q * \sigma_0$, and $f(q) < f(p)$, for each vertex p of σ_0 , implying that $\min_{p \in \tau_0} f(p) < \min_{p \in \sigma_0} f(p)$. Simplex σ_1 is a face of τ_0 different from σ_0 , which implies that q is a vertex of σ_1 , and $\min_{p \in \sigma_1} f(p) = \min_{p \in \tau_0} f(p) = f(q)$. Now $\min_{p \in \sigma_1} f(p) = \min_{p \in \tau_0} f(p) < \min_{p \in \sigma_0} f(p)$. If we continue in this way inductively, we conclude that $\min_{p \in \sigma_0} f(p) = \min_{p \in \tau_r} f(p) < \min_{p \in \sigma_0} f(p)$, a contradiction. Thus, there are no closed $EGrad f$ -paths in Σ , and $EGrad f$ is a Forman gradient vector field.

7 A Discrete Forman Function for the Extended Gradient Field

In this Section, we construct a Forman function F , which extends scalar field f , such that the discrete gradient vector field V_F of F coincides with V . We denote as $d(\gamma)$ the length of the longest V -path which starts at γ , as D the maximum of $d(\gamma)$ over all cells of Σ , and as A the minimum (absolute) difference of values of function f over all vertices of Σ .

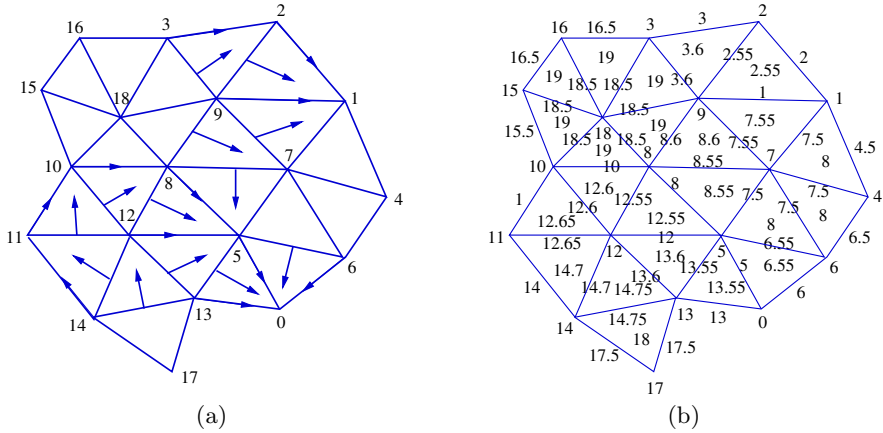


Fig. 5. Field $E\text{Grad } f$ (a) and the corresponding Forman function F (b)

Let Σ be a simplicial complex, and let f be a scalar function defined on the vertices of Σ . Let V be a discrete gradient vector field on Σ , consisting of pairs of simplexes (σ, τ) , i.e., such that $V(\sigma) = \tau = q * \sigma$, where q is a vertex of Σ . If $f(q) < f(p)$ for all vertices $p \in \sigma$, then an *extended discrete function* F on all i -simplexes γ of Σ , $0 \leq i \leq n$, given by

- (1) If γ is critical, then $F(\gamma) = \max_{p \in \gamma} f(p) + i \frac{A}{n}$,
- (2) If $V(\gamma) \neq \emptyset$, then $F(\gamma) = \max_{p \in \gamma} f(p) + i \frac{A}{n} (1 + \frac{d(\gamma)}{nD}) = \max_{p \in \gamma} f(p) + i \frac{A}{n} + i \frac{A}{n^2} \frac{d(\gamma)}{D}$,
- (3) If $\gamma = V(\sigma)$, then $F(\gamma) = F(\sigma)$,

is a Forman function, such that gradient vector field V_F of F coincides with V . For $i = 0$, an i -dimensional simplex is a vertex p of Σ , for which we have $F(p) = f(p)$. Thus, f is the restriction of F on the set of vertices of Σ . The first term, $\max_{p \in \gamma} f(p)$, relates functions f and F . The second term, $i \frac{A}{n}$, ensures that

each face τ of a simplex σ has a lower F value than σ . The third term, $i \frac{A}{n^2} \frac{d(\gamma)}{D}$, ensures that function F is non-increasing along each descending gradient path. In Figure 5, we give an example of an extended discrete function F , induced by the extended discrete field of Section 6. Here, $n = 2$, $D = 5$, and $A = 1$.

In the proof that F is a Forman function, we use the following facts:

- (a) if σ is a face of τ , then $\max_{p \in \sigma} f(p) \leq \max_{p \in \tau} f(p)$,
- (b) if σ is an i -dimensional simplex, and τ is a j -dimensional simplex of Σ ($0 \leq i < j \leq n$), then

$$i \frac{A}{n} \leq i \frac{A}{n} + i \frac{A}{n^2} \frac{d(\sigma)}{D} \leq i \frac{A}{n} + i \frac{A}{n^2} < i \frac{A}{n} + n \frac{A}{n^2} = i \frac{A}{n} + \frac{A}{n} = (i + 1) \frac{A}{n} \leq j \frac{A}{n}.$$

We consider a pair of simplexes σ and τ , such that σ is an i -simplex, τ is a j -simplex, $0 \leq i < j \leq n$, and σ is a (proper) face of τ , and we show that $F(\sigma) < F(\tau)$ if $V(\sigma) \neq \tau$. Then, the conclusion will follow for $i = j - 1$. We consider separately the cases when σ and τ belong to one of the three sets which partition the set of all simplexes of Σ , namely critical simplexes, simplexes which have a non-empty image by V , and simplexes which are in the image of V . For lack of space, we give the proof in two cases only.

(i) σ is critical, and $V(\tau) \neq \emptyset$. Then by (a) and (b)

$$F(\sigma) = \max_{p \in \sigma} f(p) + i \frac{A}{n} < \max_{p \in \tau} f(p) + j \frac{A}{n} + j \frac{A}{n^2} \frac{d(\tau)}{D} = F(\tau).$$

(ii) $V(\sigma) \neq \emptyset$, and $\tau \in ImV$.

If $V(\sigma) = \tau$, then $F(\sigma) = F(\tau)$.

If $V(\sigma) = \gamma \neq \emptyset$ and $\tau = V(\tau')$ for some $\tau' \neq \sigma$, then $F(\tau) = F(\tau')$, $\tau = q * \tau'$ for some vertex $q \in \Sigma$.

If τ' is a $(j - 1)$ -dimensional co-face of σ (if $q \notin \sigma$), then $j - 1 > i$ ($\tau \neq V(\sigma)$, thus $\tau' \neq \sigma$, and τ' is a proper co-face of σ), and by (a) and (b)

$$F(\sigma) = \max_{p \in \sigma} f(p) + i \frac{A}{n} + i \frac{A}{n^2} \frac{d(\sigma)}{D} < \max_{p \in \tau'} f(p) + (j - 1) \frac{A}{n} + (j - 1) \frac{A}{n^2} \frac{d(\tau')}{D} = F(\tau).$$

If, on the other hand, τ' is not a co-face of σ (if $q \in \sigma$), then $f(q) < f(p)$, for all $p \in \tau'$, and all other vertices of σ different from q (if they exist), are in τ' . Thus, $\max_{p \in \sigma} f(p) \leq \max_{p \in \tau'} f(p)$.

If $j - 1 > i$, then by (b)

$$F(\sigma) = \max_{p \in \sigma} f(p) + i \frac{A}{n} + i \frac{A}{n^2} \frac{d(\sigma)}{D} < \max_{p \in \tau'} f(p) + (j - 1) \frac{A}{n} + (j - 1) \frac{A}{n^2} \frac{d(\tau')}{D} = F(\tau).$$

If $j - 1 = i$, then τ' and σ are i -cells, $V(\tau') = \tau$, and σ is a face of τ , implying that $d(\tau') = d(\sigma) + 1$, and

$$F(\sigma) = \max_{p \in \sigma} f(p) + i \frac{A}{n} + i \frac{A}{n^2} \frac{d(\sigma)}{D} < \max_{p \in \tau'} f(p) + (j - 1) \frac{A}{n} + (j - 1) \frac{A}{n^2} \frac{d(\tau')}{D} = F(\tau).$$

It can be proven in a similar way in all other cases that for each pair of simplexes σ and τ , where σ is a face of τ , $F(\sigma) < F(\tau)$, unless if $V(\sigma) = \tau$, in which case $F(\sigma) = F(\tau)$. Thus, F is a Forman function on Σ .

Note that we can define another Forman function G by

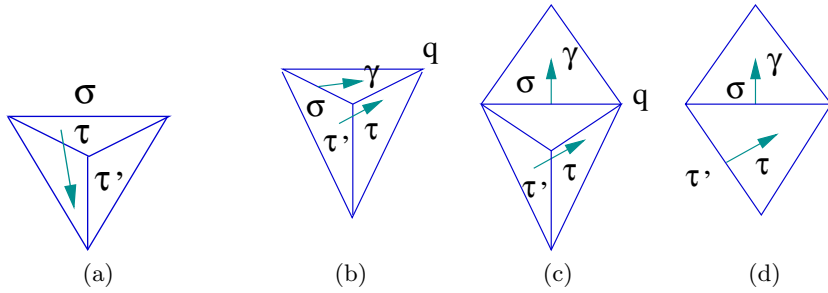


Fig. 6. (a) σ is a critical simplex, τ is a co-face of σ , and $V(\tau) \neq \emptyset$. $V(\sigma) = \gamma \neq \emptyset$, and τ is a co-face of γ , (b) $\tau = V(\tau')$ and τ' is a co-face of σ ; $\tau = V(\tau')$ and τ' is not a co-face of σ (with dimension of τ' (c) greater than, or (d) equal to, dimension of σ).

- (1) If γ is critical, then $G(\gamma) = \max_{p \in \gamma} f(p) + i \frac{A}{n} \delta'$,
- (2) If $V(\gamma) \neq \emptyset$, then $F(\gamma) = \max_{p \in \gamma} f(p) + i \frac{A}{n} \delta' + i \frac{A}{n^2} \frac{d(\gamma)}{D} \delta''$,
- (3) If $\gamma = V(\sigma)$, then $F(\gamma) = F(\sigma)$,

where $\delta' < \frac{\varepsilon}{2} \frac{1}{A}$, and $\delta'' < \frac{\varepsilon}{2} \frac{n}{A}$ for some arbitrary small positive $\varepsilon < A$. Such function $G(\sigma)$ differs by at most ε from the maximum of f over all vertices of σ .

8 Concluding Remarks

We have defined an extended form $EGrad f$ of a discrete gradient vector field $Grad f$, and we have defined a Forman function F which coincides with f on the vertices of Σ , such that the gradient field of F coincides with $EGrad f$. Our work has similarities with the work by King et al. [11], and by Gyulassy et al. [9] (when applied to a triangulated domain). The main difference is that our vector field always points in the direction of the vertex with smallest function value.

This property is not always satisfied by other approaches that construct a Forman gradient vector field V starting from a scalar function f given on vertices of Σ , as pointed out in [9, 11]. In this sense, our gradient field $EGrad f$ reflects better the behavior of the scalar field f .

Moreover, in [9], a Forman function whose gradient vector field coincides with V is not constructed. In [11], such a Forman function F , which extends scalar field f , is constructed. It satisfies an additional condition that for each simplex σ , $F(\sigma)$ is arbitrarily close to the $\max_{p \in \sigma} f(p)$. We have shown in Section 7 that the Forman function F that we construct can be re-scaled to another Forman function G satisfying the same condition.

As an application of our work, we plan to apply the Forman simplification (cancellation) operator to our extended discrete gradient field $EGrad f$ and discrete gradient field $Grad f$. In this way, we hope to reduce significantly the number of critical cells in Σ . We plan to investigate how this simplification

affects the vector fields, in order to define a multi-resolution model based on both Morse and Forman theory, and obtain a compact and topology preserving multi-resolution representation of scalar field f .

Acknowledgement. We would like to thank Emanuele Danovaro for Figure 3. This work has been partially supported by the National Science Foundation under grant CCF-0541032.

References

1. Cazals, F., Chazal, F., Lewiner, T.: Molecular Shape Analysis Based upon the Morse-Smale Complex and the Connolly Function. In: Proceedings of the nineteenth Annual Symposium on Computational Geometry, pp. 351–360 (2003)
2. Čomić, L., De Floriani, L.: Multi-Scale 3D Morse Complexes. In: International Conference on Computational Science and its Applications (ICCSA), Workshop on Computational Geometry and Applications, pp. 441–451 (2008)
3. Cousty, J., Bertrand, G., Couprie, M., Najman, L.: Collapses and Watersheds in Pseudomanifolds. In: Wiederhold, P., Barneva, R.P. (eds.) IWCIA 2009. LNCS, vol. 5852, pp. 397–410. Springer, Heidelberg (2009)
4. Danovaro, E., De Floriani, L., Mesmoudi, M.M.: Topological Analysis and Characterization of Discrete Scalar Fields. In: Asano, T., Klette, R., Ronse, C. (eds.) Geometry, Morphology, and Computational Imaging. LNCS, vol. 2616, pp. 386–402. Springer, Heidelberg (2003)
5. Danovaro, E., De Floriani, L., Vitali, M., Magillo, P.: Multi-Scale Dual Morse Complexes for Representing Terrain Morphology. In: GIS 2007: Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems, pp. 1–8. ACM, New York (2007)
6. De Floriani, L., Mesmoudi, M.M., Danovaro, E.: Smale-Like Decomposition for Discrete Scalar Fields. In: Proceedings International Conference on Pattern Recognition, ICPR (2002)
7. Forman, R.: Combinatorial Vector Fields and Dynamical Systems. *Mathematische Zeitschrift* 228, 629–681 (1998)
8. Forman, R.: Morse Theory for Cell Complexes. *Advances in Mathematics* 134, 90–145 (1998)
9. Gyulassy, A., Bremer, P.T., Hamann, B., Pascucci, V.: A Practical Approach to Morse-Smale Complex Computation: Scalability and Generality. *Transactions on Visualization and Computer Graphics* 14(6), 1619–1626 (2008)
10. Jerše, G., Mramor Kosta, N.: Ascending and descending regions of a discrete morse function. *Comput. Geom. Theory Appl.* 42(6-7), 639–651 (2009)
11. King, H., Knudson, K., Mramor, N.: Generating Discrete Morse Functions from Point Data. *Experimental Mathematics* 14(4), 435–444 (2005)
12. Lewiner, T., Lopes, H., Tavares, G.: Applications of Forman’s Discrete Morse Theory to Topology Visualization and Mesh Compression. *Transactions on Visualization and Computer Graphics* 10(5), 499–508 (2004)
13. Matsumoto, Y.: An Introduction to Morse Theory, *Translations of Mathematical Monographs*, vol. 208. American Mathematical Society, Providence (2002)
14. Milnor, J.: Morse Theory. Princeton University Press, New Jersey (1963)

ACCORD: With Approximate Covering of Convex Orthogonal Decomposition

Mousumi Dutt¹, Arindam Biswas¹, and Partha Bhowmick²

¹ Department of Information Technology,
Bengal Engineering and Science University, Shibpur, Howrah, India
{duttmousumi, barindam}@gmail.com

² Department of Computer Science and Engineering,
Indian Institute of Technology, Kharagpur, India
bhowmick@gmail.com

Abstract. A fast and efficient algorithm to obtain an orthogonally convex decomposition of a digital object is presented. The algorithm reports a sub-optimal solution and runs in $O(n \log n)$ time for a hole-free object whose boundary consists of n pixels. The approximate/rough decomposition of the object is achieved by partitioning the inner cover (an orthogonal polygon) of the object into a set of orthogonal convex components. A set of rules is formulated based on the combinatorial cases and the decomposition is obtained by applying these rules while considering the concavities of the inner cover. Experimental results on different shapes have been presented to demonstrate the efficacy, elegance, and robustness of the proposed technique.

Keywords: convex component; convex decomposition; image analysis; polygon decomposition; rectangular component.

1 Introduction

Polygons are frequently used by practitioners to solve geometric problems related with image processing and computer vision [6, 7, 21–23, 27, 28]. Computational-geometric problems on general polygons are solved usually by decomposing them into simpler components, solving the problem for each component using a specialized algorithm, and then combining the partial solutions. A polygon can be decomposed into rectangular components, convex components, star-shaped components, monotone components, etc. Polygon decomposition has many applications in theory and practice [4, 10, 25–27].

Polygon decompositions can be classified based on the interrelation among decomposed components. If the components do not overlap except at their boundaries, then the decomposition is termed as a *partition*. If overlapping pieces are allowed, then it is called a *cover*. Decomposing a polygon into simpler components can be performed with or without addition of extra vertices, which makes the problem complex but leads to fewer components. A decomposition must be minimal in some sense. Some applications use decomposition of a polygon into

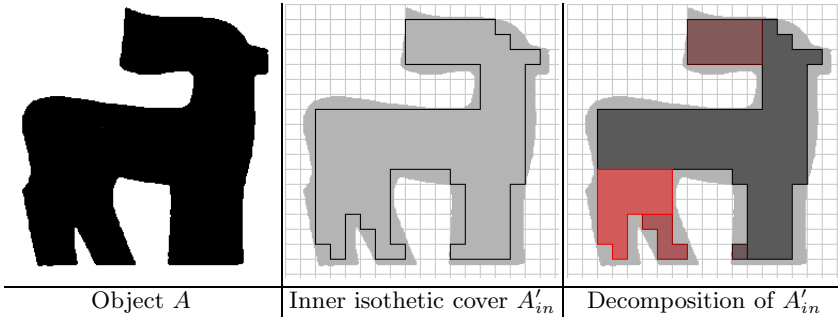


Fig. 1. A sample digital object and its orthogonal decomposition by our algorithm

minimum number of components, and some other applications use decomposition that minimizes the total length of internal edges [11]. In our work, decomposition is performed using the notion of partitioning.

Many works have been done on decomposition in general domain as well as in orthogonal domain. In general domain, decomposition of a polygon with holes into minimum number of components is NP-hard [5, 15, 17, 20], whether allowing or disallowing Steiner points. Allowing Steiner points, an $O(n \log n)$ -time approximation algorithm under minimum edge-length criteria was stated in [12]. No optimal algorithm is known for decomposing hole-containing polygons when Steiner points are allowed; in [8, 16], it has been shown that such problems are NP-hard. In 3D domain also, several works have been done, and in [1, 13, 14] some approximation algorithms are given, since exact convex decomposition is NP-hard. Various applications of decomposition problems may be seen in [2, 7, 16, 18, 19].

Interestingly, the problems which are NP-hard for general polygons, may become tractable when restricted to orthogonal domain. An $O(n^2)$ -time algorithm to cover a simple orthogonal polygon with minimum number of orthogonally convex polygons is given in [24], where polygons are classified and decomposed based on dent diagrams. Another $O(n^2)$ -time algorithm in [9] gives the solution to the problem of covering a horizontally convex orthogonal polygon with minimum number of orthogonally convex polygons and with the minimum number of orthogonal star-shaped polygons. But, our algorithm allows all kinds of orthogonal polygon without holes as input. Our work is focused on the decomposition of a given orthogonal polygon (i.e., polyomino) without holes (which acts as the tight *inner isothetic cover* of a hole-free digital object) into a sub-optimal¹ set of *orthogonally convex components* (OCC, or, *hv-convex polyominoes*) such that (c1) each OCC is orthogonal with all its vertices as grid points and (c2) no two OCC overlap each other except at their boundaries. Condition c1 implies that extra vertices are allowed only in the form of grid points. Figure 1 illustrates a digital object A , its inner isothetic cover (for $g = 13$), and the decomposed

¹ Exhaustive experimentation (Sec. 4) shows that our algorithm frequently produces optimal solution.

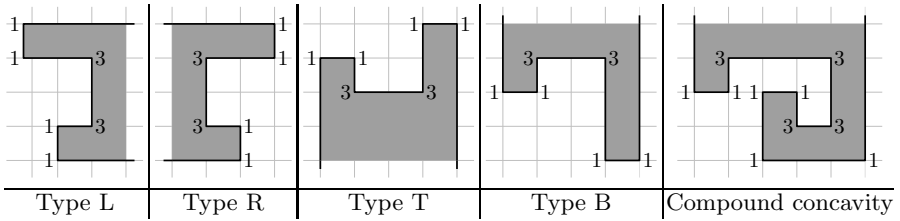


Fig. 2. Different types of concavities

set of OCC (which needed no extra vertex). It should be mentioned here that, to the best of our knowledge, there exists no proof till date to show whether *partitioning* an orthogonal polygon into a minimal set of OCCs can be done in polynomial time.

2 Preliminaries

A given *digital object*, A , is first imposed on an orthogonal *grid*, G . The grid consists of a set of equi-spaced horizontal lines and a set of equi-spaced vertical lines, their vertical/horizontal spacing being termed as the *grid spacing*, g . Using the algorithm TIPS [3], we obtain (the ordered set of vertices of) the *inner isothetic cover*, A'_{in} , which is the maximum-area orthogonal polygon inscribing A such that the vertices of A'_{in} are *grid points* (points of intersection among horizontal and vertical grid lines).

2.1 Storing the Vertices

The vertices of A'_{in} are dynamically inserted in the circular doubly-linked list, L , and simultaneously in two temporary tables, H_x and H_y , during the construction of A'_{in} . Each table is structured like a hash table without any hash key, and contains all slots from the minimum to the maximum grid coordinate. In H_x (H_y), a vertex $v_i(x_i, y_i)$, is stored in the slot of x_i (y_i). The size of each slot x_i (y_i) of H_x (H_y) is dynamically allocated depending on the number of vertices having abscissa x_i (ordinate y_i); the concerned number of vertices is obtained by lexicographic sorting of L to two temporary lists, L'_x (x - and y -coordinates as the respective primary and secondary keys) and L'_y (y - and x -coordinates as the respective primary and secondary keys). L'_x (L'_y) is also used to prepare the slots of H_x (H_y) in $O(n/g)$ time, since $O(n/g)$ is the number of vertices (explained in Sec. 3.5), n being the number of border pixels of A , and g , the grid size. For each vertex v_i , L contains its coordinates, *type* ($= 1$ when the internal angle at v_i is $\theta_i = 90^\circ$ and 3 when $\theta_i = 270^\circ$), and *id* of the cell incident at v_i with partial object occupancy only if *type* $= 3$; for *type* $= 1$, *id* is set to 0 . (Example: With *type* $= 3$, if the top-right cell is partially occupied by the object A , then *id* $= 1$; if it is the top-left one, then *id* $= 2$, and so on.)

In L , the component number k_i (initialized as '1') corresponding to each vertex v_i , is also stored. When a component is extracted during decomposition, the *component count* k is increased and k_i of each vertex of the extracted

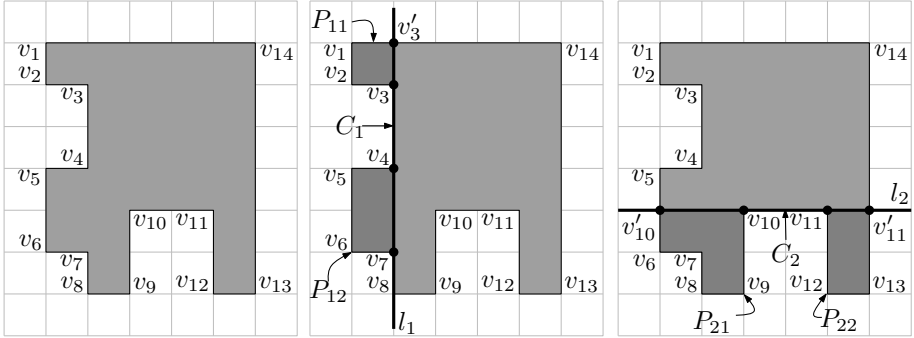


Fig. 3. Start and terminal vertices of $P_{11}, P_{12}, P_{21}, P_{22}$ w.r.t. C_1 and C_2

component is reset to k . During the construction of A'_{in} , a linked list L_c is created, which contains all the concavities in the order as they appear in L . (Two or more consecutive Type 3 vertices creates a concavity [Fig. 2].) Each entry in L_c contains the coordinates of two consecutive Type 3 vertices of the concerned concavity along with concavity type and concavity number. A vertex pattern of “1331” may occur in four possible *simple concavities*, namely, Type L (left), Type R (right), Type T (top), and Type B (bottom) [Fig. 2], determined from the Type 3 vertex *ids* stored in L . Three or more consecutive Type 3 vertices form a *compound concavity*, which are stored as simple concavities in L_c , each consisting of two consecutive Type 3 vertices and these simple concavities are assigned the same concavity number in L_c .

2.2 Identifying the Sub-polygons of a Concavity

If a horizontal/vertical line (as the case may be) l_i is drawn along the (simple) concavity C_i , then the polygon is divided into two sub-polygons lying on one side of l_i and the the main polygon on other side (Fig. 3). l_i is called *concavity line* passing through the consecutive Type 3 vertices of C_i . If one of the two sub-polygons is extracted as an OCC, then C_i is resolved. Each sub-polygon has at least two points on the concavity line, l_i , named as the *start vertex* (Type 3) and the *terminal vertex*. In Fig. 3, for concavity C_1 , the respective sub-polygons P_{11} and P_{12} have $s_{11} = v_3$ and $s_{12} = v_4$ as start vertices, and $t_{11} = v'_3$ and $t_{12} = v_7$ as terminal vertices. The sub-polygon appearing before (after) the concavity is described in the clockwise (counterclockwise) manner.

Terminal vertex of a sub-polygon is found using H_x and H_y , as all vertical edges are stored in H_x and horizontal edges in H_y . For the polygon shown in Fig. 3, $L = \langle v_1, v_2, \dots, v_{14} \rangle$. For C_1 (Type L), we obtain the slot of H_x containing the start vertices v_3 and v_4 . As v_7 appears next to v_4 , v_7 becomes the terminal vertex of P_{12} . However, as no vertex exists in H_x before v_3 , the horizontal edge (v_1, v_{14}) lying above v_3 is detected from H_y . The x -coordinate of v_3 lies between those of v_1 and v_{14} , whence the y -coordinate of the terminal vertex of P_{11} equals that of v_1 or v_{14} . For C_2 (Type B) or a Type T/Type R concavity, the terminal vertices are detected in a similar manner.

3 Rules for Decomposition

The decomposition is carried out using L_c (Sec. 2.1). To resolve each concavity, apparently one convex component should be extracted. For example, in Fig. 3, extraction of P_{11} resolves C_1 and extraction of P_{21} resolves C_2 , which, however, does not yield an optimal solution. Hence, rules are always applied to a pair of concavities at a time. The (sub-)optimality is obtained as explained next.

3.1 Two Simple, Orthogonal, Consecutive Concavities

If the lines of concavity l_1 and l_2 , corresponding to C_1 and C_2 , are orthogonal and intersect at v , and no sub-polygon of any concavity contains the other concavity in full, then extraction of only one sub-polygon resolves both C_1 and C_2 (Fig. 4(a-d)). The combined type of C_1 and C_2 may be LB, BR, RT, or TL. By traversing from s_{12} to $v = t_{12}$, the component is extracted to resolve two concavities at a time. We have the following cases:

- $v \in \{s_{11}, s_{12}, s_{21}, s_{22}\}$ (Fig. 4(a)): v is a vertex present in L .

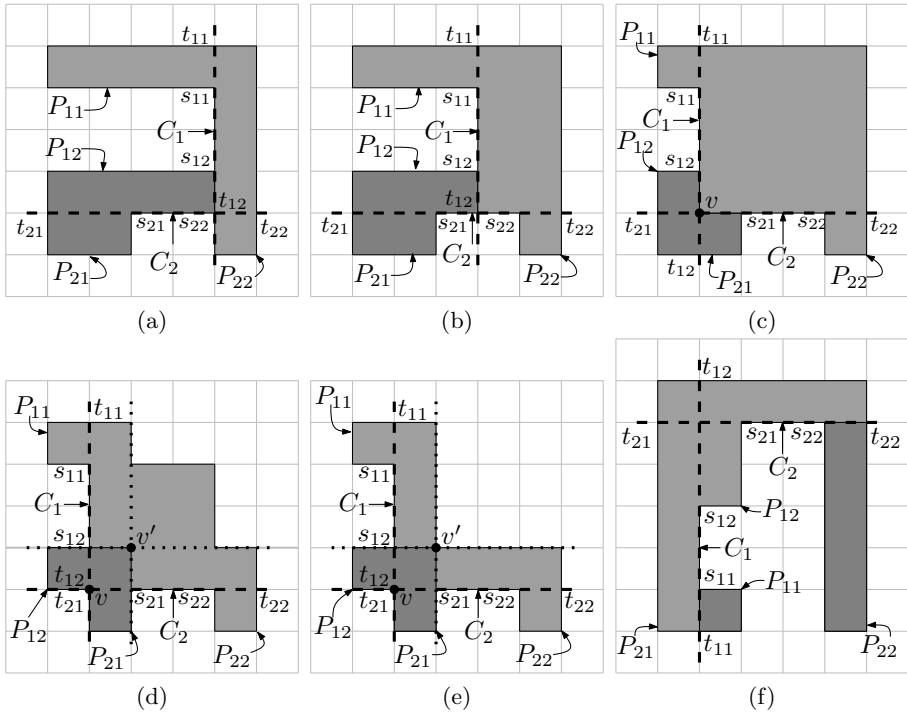


Fig. 4. Rules of decomposition for $l_1 \perp l_2$. (There also exist rotational versions of the above cases—dealt in similar ways.)

- v lies on the edge (s_{11}, s_{12}) or (s_{21}, s_{22}) (Fig. 4(b)): v is inserted appropriately in L (using H_x and H_y).
- v lies not on the boundary but inside A'_{in} (Fig. 4(c)): v is inserted in L (as above).
- v lies on (the boundary of) or outside A'_{in} (Fig. 4(d,e)): Find the point of intersection, v' , between the line perpendicular to l_1 and passing through s_{12} and the line perpendicular to l_2 and passing through s_{21} . If v' lies inside A'_{in} , then the component is extracted by anticlockwise traversal from s_{12} down to v' and ending at s_{12} (Fig. 4(d)). If v' lies on or outside A'_{in} , then P_{11} is first extracted to solve C_1 and then P_{22} to solve C_2 (Fig. 4(e)).
- If C_1 (C_2) lies entirely in one sub-polygon, say P_{21} , corresponding to C_2 (C_1), then both P_{11} and P_{22} are extracted (Fig. 4(f)).

3.2 Two Simple, Parallel, Consecutive Concavities

If the projection of the edge (s_{11}, s_{12}) on l_2 (or of the edge (s_{21}, s_{22}) on l_1) lies on or inside A'_{in} , then extraction of one sub-polygon resolves both C_1 and C_2 (Fig. 5(a-d)). Otherwise, P_{11} and P_{21} are extracted to resolve the respective concavities, C_1 and C_2 (Fig. 5(e, f): extracted sub-polygons shown in deep gray).

3.3 Compound Concavities

When there is a series of $t(> 2)$ consecutive Type 3 vertices, then it is broken into $t - 1$ simpler concavities, each consisting of two consecutive Type 3 vertices. For example, in Fig. 6 there are three pairs L_1T_1 , R_1B_1 , and L_2T_2 , which are solved in three steps.

3.4 Demonstration

In Fig. 7(a), the concavities are numbered in the order in which they are stored in L_c . The first concavity in L_c is C_1 , whose next and previous concavities are C_2 and C_{10} , respectively. As the concavity lines of C_1 and C_{10} are parallel, the rules of Sec. 3.2 are applied to extract the OCC (shown in deep gray in Fig. 7(c)). The second among the three consecutive Type 3 vertices (for C_1 and C_2) is included in the extracted OCC. Thus, three concavities are solved at a time (C_1 , C_2 , and C_{10}). L_c gets updated, and C_3 and C_4 are first checked, but no rules can be applied on them. So, C_3 is checked with its previous concavity in L_c , i.e., C_9 . They are at 180° , and the OCC is extracted (Fig. 7(d)). Next pair is C_4 and C_5 , which does not admit a single OCC extraction; so, C_4 is checked with its previous, i.e., C_8 , but they also do not. Hence, one more backward traversal is made to test C_4 and C_7 . Their concavity lines are parallel, and the OCC is extracted according to the rules stated in Sec. 3.2 (shown in Fig. 7(e)). In this manner, other concavities are also removed, to obtain the final result (Fig. 7(f)). The total number of OCC is $k = 7$ against the total number of simple concavities as $c = 10$.

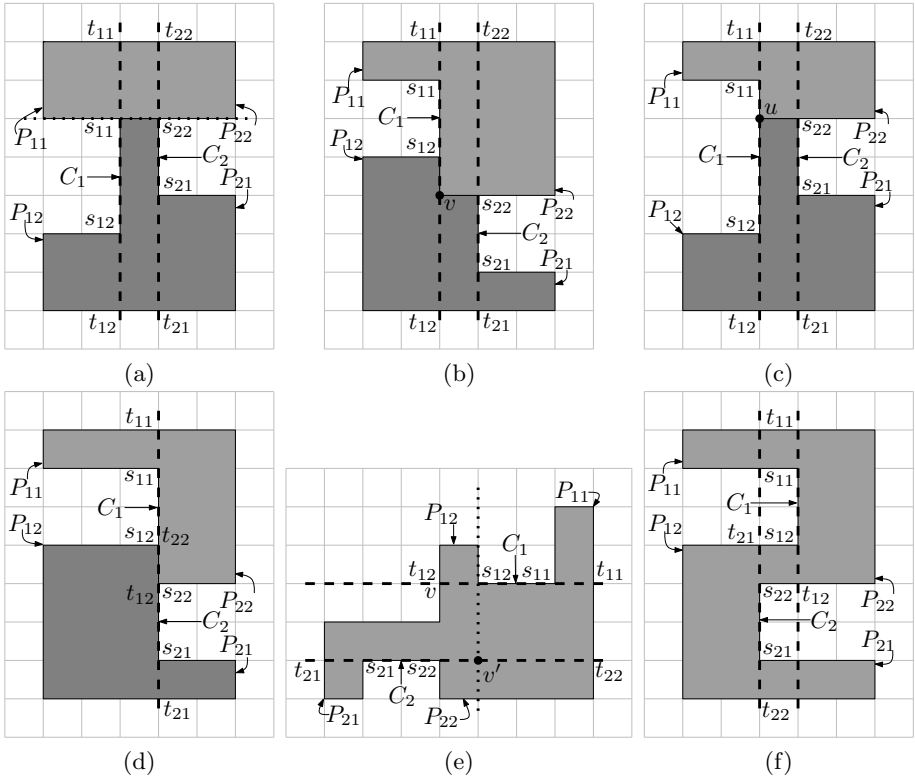


Fig. 5. Rules of decomposition for l_1 parallel to l_2 . (There also exist rotational/flop versions of the above cases—dealt in similar ways.)

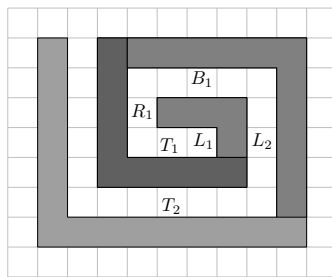


Fig. 6. Decomposition of a compound concavity

3.5 Time Complexity

The algorithm first constructs the inner isothetic cover A'_{in} , vertex list L , concavity list L_c , and the two hash tables (H_x and H_y). All this needs $O(n)$ time,

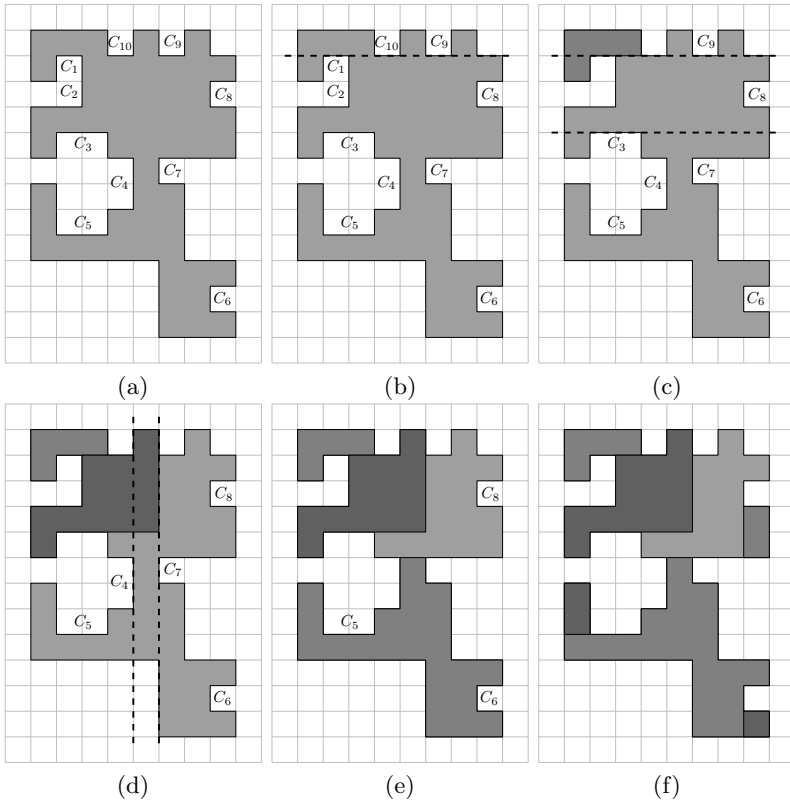


Fig. 7. A demonstration of getting OCC by our algorithm

where n is the pixels constituting the contour of the object A , which is considered here as a connected component without holes. This follows from the combined fact that the intersection of each grid edge with A is checked in $O(g)$ time, and the number of grid points visited is bounded by $O(n/g)$. All grid points lying on A'_{in} are inserted in L in $O(1)$ time. All the concavities are inserted in L_c in $O(n)$ time. For preparation of H_x and H_y from L'_x and L'_y (Sec. 2.1), total computational cost is $O(n \log n)$. Hence, the total time complexity for Stage 1 is $O(n \log n)$.

In Stage 2, traversal of L_c needs $O(n)$ time. Searching in H_x and H_y is performed in this stage, which takes $O(\log n)$ computational time per search. This follows from the fact that we apply binary search in each slot, which contains at most $O(n)$ elements in the dynamically-allocated array form (Sec. 2.1). There would be $O(n)$ such searches, consuming $O(n \log n)$ time in total. Insertion of extra grid points in L may take $O(n \log n)$ time in worst case. Deletion of each node from L_c requires $O(1)$ time, thereby consuming $O(n)$ time in total. Hence, the total worst-case time complexity for Stage 2 also is $O(n \log n)$, wherefore the overall time complexity of the entire algorithm becomes $O(n \log n)$.



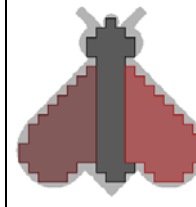
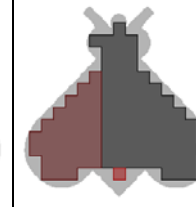



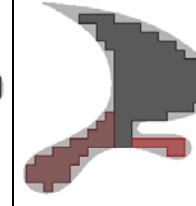


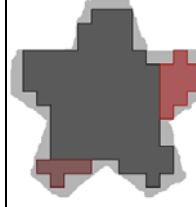
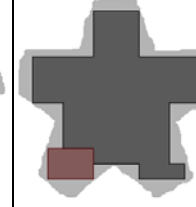

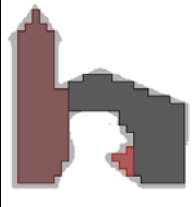
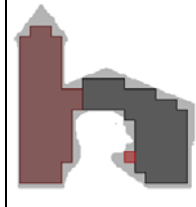
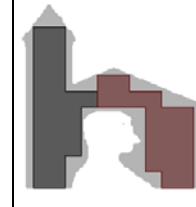

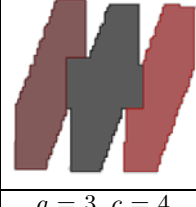
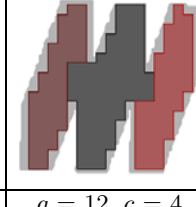
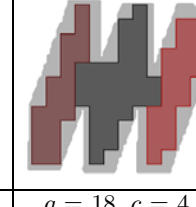
			
Logo 416	$g = 7, c = 6,$ $k = 5$	$g = 12, c = 4,$ $k = 3$	$g = 15, c = 3,$ $k = 3$
			
Logo 111	$g = 1, c = 3,$ $k = 3$	$g = 5, c = 3,$ $k = 3$	$g = 11, c = 2,$ $k = 3$
			
Logo 5	$g = 10, c = 3,$ $k = 3$	$g = 17, c = 4,$ $k = 3$	$g = 19, c = 2,$ $k = 2$
			
Logo 230	$g = 9, c = 3,$ $k = 3$	$g = 13, c = 3,$ $k = 3$	$g = 20, c = 2,$ $k = 2$
			
Logo 294	$g = 3, c = 4,$ $k = 3$	$g = 12, c = 4,$ $k = 3$	$g = 18, c = 4,$ $k = 3$

Fig. 8. Experimental results of orthogonal convex decomposition for various grid sizes








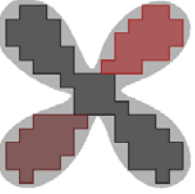










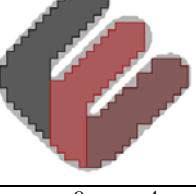
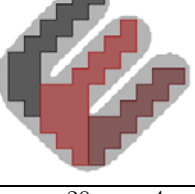
			
Logo 294	$g = 3, c = 2,$ $k = 3$	$g = 7, c = 2,$ $k = 3$	$g = 20, c = 2,$ $k = 3$
			
Logo 364	$g = 1, c = 4,$ $k = 3$	$g = 10, c = 4,$ $k = 3$	$g = 17, c = 4,$ $k = 3$
			
Logo 471	$g = 5, c = 2,$ $k = 2$	$g = 8, c = 2,$ $k = 2$	$g = 12, c = 2,$ $k = 2$
			
Logo 571	$g = 14, c = 2,$ $k = 2$	$g = 15, c = 2,$ $k = 2$	$g = 19, c = 3,$ $k = 3$
			
Logo 7	$g = 3, c = 4,$ $k = 3$	$g = 9, c = 4,$ $k = 3$	$g = 20, c = 4,$ $k = 3$

Fig. 9. Another set of results of orthogonal convex decomposition for various grid sizes

4 Results and Conclusion

We have implemented the algorithm in C in Linux Fedora Release 7, Kernel version 2.6.21.1.3194.fc7, Dual Intel Xeon Processor 2.8 GHz, 800 MHz FSB. It is tested on several datasets containing various digital images of different shapes and forms. The decomposition takes place on inner cover of the images. In Figs. 8 and 9, results of several typical objects are given for various grid sizes. The different OCCs are marked by different colors. The total number of decomposed OCCs depends on the number of concavities and also on their mutual configurations and orientations. The count k of OCC for a given digital object also depends on the grid size (required while extracting the inner cover A'_{in} of the object A), since the number of concavities (c) increases for lower grid sizes.

The proposed algorithm is designed in a manner so as to decompose arbitrary digital objects into *orthogonally convex components* (OCC), which might be useful for shape analysis. The algorithm can decompose a hole-free orthogonal polygon into a sub-optimal set of OCCs in $O(n \log n)$ time by considering all the concavities of the polygon. The steps of the algorithm are based on the rules designed out of the combinatorial possibilities. Exhaustive experimentation has been performed to verify its efficiency and robustness, and the results show the decompositions, which are mostly optimal.

References

1. Aguilera, A., Ayala, D.: Faster ASV Decomposition for Orthogonal Polyhedra, Using the Extreme Vertices Model (EVM). In: WSCG 2000, pp. 60–67 (2000)
2. Asano, T., Asano, T., Imai, H.: Partitioning a Polygonal Region into Trapezoids. JACM 33, 290–312 (1986)
3. Biswas, A., Bhowmick, P., Bhattacharya, B.B.: TIPS: On finding a tight isothetic polygonal shape covering a 2D object. In: Kalviainen, H., Parkkinen, J., Kaarna, A. (eds.) SCIA 2005. LNCS, vol. 3540, pp. 930–939. Springer, Heidelberg (2005)
4. Chazelle, B.: Approximation and Decomposition of Shapes. In: Schwartz, J.T., Yap, C.K. (eds.) Advances in Robotics 1: Algorithmic and Geometric Aspects of Robotics, pp. 145–186. Lawrence Erlbaum Associates, Hillsdale (1987)
5. Chazelle, B., Dobkin, D.: Decomposing a polygon into its convex parts. In: Proc. STOC, pp. 38–48 (1979)
6. Feng, H.Y.F., Pavlidis, T.: Decomposition of Polygons Into Simpler Components: Feature Generation for Syntactic Pattern Recognition. IEEE Trans. Computers 24, 636–650 (1975)
7. Ferrari, L., Sankar, P.V., Sklansky, J.: Minimal Rectangular Partitions of Digitized Blobs. CVGIP 28, 58–71 (1984)
8. Keil, J.M.: Decomposing a Polygon into Simpler Components. PhD thesis, Univ. of Toronto, Canada (1983)
9. Keil, J.M.: Minimally Covering a Horizontally Convex Orthogonal Polygon. In: Proc. SoCG, pp. 43–51 (1986)
10. Keil, J.M., Sack, J.R.: Minimum Decompositions of Polygonal Objects. In: Toussaint, J.T. (ed.) Computational Geometry, Netherlands, pp. 197–216 (1985)

11. Klincsek, G.T.: Minimal Triangulations of Polygonal Domains. *Discrete Math.* 9, 121–123 (1980)
12. Levkopoulos, C., Lingas, A.: Bounds on the Length of Convex Partition of Polygons. In: Joseph, M., Shyamasundar, R.K. (eds.) *FSTTCS 1984*. LNCS, vol. 181, pp. 279–295. Springer, Heidelberg (1984)
13. Lien, J.-M., Amato, N.M.: Approximate convex decomposition of polygons. In: *Proc. SoCG*, pp. 17–26 (2004)
14. Lien, J.-M., Amato, N.M.: Approximate convex decomposition of polygons. *CGTA* 35, 100–123 (2006)
15. Lingas, A.: The power of non-rectilinear holes. In: *Proc. ICALP*. LNCS, vol. 140, pp. 369–383 (1982)
16. Lingas, A., Pinter, R., Rivest, R., Shamir, A.: Minimum Edge Length Partitioning of Rectilinear Polygons. In: *Proc. 20th Allerton Conf. Commun. Control Comput.*, pp. 53–63 (1982)
17. Lingas, A., Soltan, V.: Minimum Convex Partition of a Polygon with Holes by Cuts in Given Directions. In: *Proc. ISAAC*, vol. 1178, pp. 315–325 (2006)
18. Nahar, S., Sahni, S.: Fast Algorithm for Polygon Decomposition. *IEEE Trans. CAD* 7, 473–483 (1988)
19. Ohtsuki, T.: Minimum Dissection of Rectilinear Regions. In: *Proc. IEEE Intl. Symp. Circuits & Systems*, pp. 1210–1213 (1982)
20. O’Rourke, J., Supowit, K.J.: Some NP-Hard Polygon Decomposition Problems. *IEEE Trans. Information Theory* 29, 181–190 (1983)
21. Pavlidis, T.: Shape Discrimination. In: Fu, K.S. (ed.) *Syntactic Pattern Recognition*, NY, pp. 125–145 (1977)
22. Pavlidis, T.: *Structural Pattern Recognition*. Springer, Berlin (1977)
23. Pavlidis, T.: Survey: A Review of Algorithms for Shape Analysis. *CGIP* 7, 243–258 (1978)
24. Reckhow, R.A., Culberson, J.: Covering a Simple Orthogonal Polygon with a Minimum Number of Orthogonally Convex Polygons. In: *Proc. SoCG*, pp. 43–51 (1986)
25. O’Rourke, J.: *Art Gallery Theorems and Applications*. Oxford University Press, NY (1987)
26. Shermer, T.C.: Recent Results in Art Gallery. *Proc. of the IEEE* 80(9), 1384–1399 (1992)
27. Toussaint, G.T.: Pattern Recognition and Geometrical Complexity. In: *Proc. ICPR*, pp. 1324–1347 (1980)
28. Waco, D.L., Kim, Y.S.: Geometric Reasoning for Matching Features Using Convex Decomposition. In: *Proc. ACM Symp. Solid & Physical Modeling*, pp. 323–332 (1993)

Measures for Surface Comparison on Unstructured Grids with Different Density

Natalia Dyshkant

Lomonosov Moscow State University,
Faculty of Computational Mathematics and Cybernetics,
Leninskie gory, CMC MSU, 119991 Moscow, Russian Federation
`natalia.dyshkant@gmail.com`

Abstract. We consider the problem of surface comparison given as spatial point clouds that can be explicitly projected onto a plane. This problem can be reduced to comparison of mesh functions of two variables given on different grids. A general case when both grids are unstructured and have different density is of interest. A measure to compare such functions that allow to estimate difference on areas with nodes from both grids and an algorithm to compute it are proposed. Estimation for computational complexity of the algorithm is presented. Computing experiments on real data (3d face models) were carried out.

Keywords: discrete surface model, metrics for surface comparison, unstructured grid, Delaunay triangulation, minimum spanning tree, 3d face image.

1 Introduction

With the rapid progress of modern 3d scanning technologies [1], objects' surfaces can be routinely acquired as discrete surface models, which consist of clouds of points with 3d coordinates. Spatial objects' shape can be considered as a set of *schlicht* surfaces, i.e., such that can be uniquely projected onto a plane (see Fig. 1). In this paper, new measures for comparison of such surfaces and effective methods to compute them are devised.

Exact and computationally efficient algorithms for computing disparity measures between surfaces are required in many applications of image analysis and computer graphics. Surface comparison and matching methods are needed to solve problems of surface classification, registration, reconstruction by its separate fragments, etc.

Raw *schlicht* surface data acquired by 3d scanner can be considered as a discrete function defined at nodes of (generally speaking) *unstructured* grid. This means that nodes of a grid can be spread randomly, i.e., there is no ordered structure in the grid.

There are two main presentation methods for modelling of *schlicht* surfaces: definition of surface on structured (uniform) and unstructured grids (see Fig. 1).

The first method is tedious in case of requirement for sufficiently detailed presentation of surface. Besides, a selection problem of cell size of a grid appears. The size should be optimal to get adequate accuracy of surface approximation for concrete applied problem. In case of unstructured grids, surface approximation quality is higher. At the same time it is required to introduce and design more complex measures for surface comparison given on *different* grids and processing algorithms.

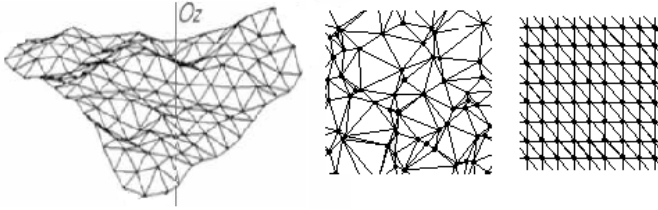


Fig. 1. Examples of schlicht surface (on the left), unstructured grid (in the middle) and uniform (structured) grid (on the right)

At the first stage of the most proposed approaches for surface comparison two source unstructured grids are transformed to a general uniform grid [2]. Then a measure between two surfaces is computed. At the transformation stage interpolation and extrapolation of source data are used. Therefore surface approximation quality decreases. A lot of methods are too computationally intensive for achievement of adequate accuracy. In this paper, we propose the approach, which saves given nonregularity of grids.

In [3] the author proposed a measure for comparison of functions, which will be discussed further. The proposed algorithm to compute the measure used Delaunay triangulations of each point clouds, general Delaunay triangulation constructed for both clouds, function interpolation on basis of localization of triangulations in each other and function comparison on single cells of general triangulation. Localization was implemented on basis of minimum spanning trees. In this paper, we concentrate on modifications of this measure for case when source grids are not only unstructured but also have different density and designing an effective algorithm to compute new measures.

In practise, such measures are needed for comparison of surfaces acquired by 3d scanners of different accuracy: there are applications for 3d face model matching or matching of 3d face model of person and his 3d jaw models in orthodontics [4]. Jaw models are acquired by special orthodontic scanner of very high accuracy, 3d scanner for acquisition of face models is of less accuracy.

This paper is organized as follows. A review of surface comparison problem, formalized problem statement and the proposed measures are given in section 2. The proposed algorithm for measure computing and evaluation of its complexity are given in section 3. Section 4 presents experimental results for comparison of 3d face surfaces. Conclusion is given in section 5.

2 Surface Comparison Problem

Though there has been a great deal of progress in 3d object comparison field, computationally efficient and accurate surface comparison is still an urgent and unsolved problem.

2.1 Related Work

A large body of research addresses not to discrete surface model but to initial continuous surface representation or definition of surface on a uniform grid [5], [6]. In [5] measure for comparison of surfaces f_1, f_2 defined on grids g_1, g_2 was introduced as the following:

$$\rho(f_1, f_2) = \max_{g_1[i] \in g_1} \min_{g_2[j] \in g_2} d(g_1[i], g_2[j]),$$

Some researches proposed methods of referencing or comparison by descriptors. For example, in [7] the objects' comparison problem was reduced to graph comparison problem. Vertices of graph represent separate fragments of object's surface and edges represent information about their connectivity.

In [8] the problem of surfaces given on different point sets was considered. Distance from a point of one surface to the other one was computed along a normal to the nearest spline for this point. Comparison measure based on computing of distance difference along surface normals is interesting as it doesn't require transformation of source grids to the common one.

2.2 Mathematical Problem Statement

A 3d schlicht surface S given as a point cloud $\{(x_i, y_i, z_i)\}_{i=1}^N$ can be considered as the one-valued function $z = F(x, y)$ defined on the discrete set $\{(x_i, y_i)\}_{i=1}^N$. We construct plane Delaunay triangulation of this set. *Triangulated surface model* consists of spatial triangles defined by the values z_i at nodes (x_i, y_i) of this triangulation. Such piecewise linear model interpolates the initial surface S .

Definition 1. *A finite point set $G : \{(x^i, y^j) \in \mathbb{R}^2 | i = 1, \dots, N\}$, $N \geq 3$ is called a nonuniform two-dimensional grid.*

Suppose R is a rectangle in \mathbb{R}^2 , \mathcal{G} is the set of nonuniform two-dimensional grids contained in R , \mathcal{F} is the set of single-valued functions given on grids from \mathcal{G} .

Consider the following problem statement.

Let S_1, S_2 be two schlicht surfaces defined by discrete functions $F_1, F_2 \in \mathcal{F}$ at the nodes of grids $G_1, G_2 \in \mathcal{G}$, respectively, $N_1 = |G_1|, N_2 = |G_2|$. It is required to introduce measures for comparison of surfaces S_1, S_2 and design an approach to compute them.

Note that two surfaces are given on two *different* grids and it is not possible to compute a measure between them directly. So the basic idea of the proposed approach is to fill "missing" values of each surface at the nodes of the other grid.

Let \hat{F} be a continuous piecewise linear function in \mathbb{R}^2 . We say that \hat{F} approximates F at G if $\hat{F}(x, y) \equiv F(x, y)$ for all $(x, y) \in G$. By $\hat{\mathcal{F}}$ denote the set of continuous functions in \mathbb{R}^2 that approximates functions from \mathcal{F} .

Denote by T_1 and T_2 the Delaunay triangulations constructed on grids G_1 and G_2 , respectively. The Delaunay triangulation constructed on the union of two grids $G = G_1 \cup G_2$ is called the *general* (or *united*) Delaunay triangulation and is denoted by T .

2.3 Proposed Measures

Let ρ be a function in \mathcal{F} such that the following conditions hold: $\forall F_1, F_2, F_3 \in \mathcal{F} : \rho(F_1, F_2) \geq 0, \rho(F_1, F_2 + F_3) \leq \rho(F_1, F_2) + \rho(F_1, F_3)$. We shall say that ρ is a difference functional of two functions (schlicht surfaces).

Consider a function $\mu(x, y)$ that defined *weight* of difference between two surfaces at a certain point with coordinates (x, y) in accordance with significance of function similarity in the region contained this point. Let A, B, C be nodes of the united grid G . By definition, put:

$$V_\mu(A, B, C, F_1, F_2) = \iint_{\Delta ABC} |\hat{F}_1(x, y) - \hat{F}_2(x, y)| \mu(x, y) \, dx dy. \tag{1}$$

In [3], the author proposed the following measure:

$$\rho_{V_\mu}(F_1, F_2) = \sum_{\Delta ABC \in T} V_\mu(A, B, C, F_1, F_2) / S_{\Delta ABC}, \tag{2}$$

where weighted difference volume is summarised over all triangles ΔABC of the general triangulation and each summand is normalised by the area of triangle $S_{\Delta ABC}$.

By introducing of function $\mu(x, y)$ on surfaces the defined measure can be adapted for concrete applied problems. For example, in case of face models' comparison, μ can have greater value on regions where skin is close with a skull than on another regions with more tissues.

In fact, measure (1)-(2) is an analog of L_1 -metric for interpolated source functions \hat{F}_1 and \hat{F}_2 .

In case $\mu(x, y)$ is the identical function, similarity of all surface patches allows with equal weight. But objective nature of data is that source grids are not always uniform. There often exists such regions, which have sufficiently great area and consist of nodes from one of the grids only.

Let V be a value of V_μ for $\mu(x, y) \equiv 1$; suppose $\rho_V(F_1, F_2)$ is measure (2) for condition of $V_\mu = V : V = V_\mu|_{\mu \equiv 1}, \rho_V = \rho_{V_\mu}|_{V_\mu = V}$.

The defect of ρ_V is that it allows both type of regions where nodes from two grids are mixed with one another and where one of the grids is more thick than the other using the same weight. Such measure can be successfully used in case when nodes of both grids are distributed uniformly with roughly equal density.

In the considered case, when grids are unstructured and have different density this defect is revealed. To get out from it we propose a measure that takes in account only representative data, i.e., regions, where nodes of both grids are concentrated:

$$\rho_{\delta V}(f_1, f_2) = \sum_{\substack{\Delta ABC \in T: \\ \Delta ABC \notin T_1, \\ \Delta ABC \notin T_2}} V(A, B, C, f_1, f_2) / S_{\Delta ABC}. \tag{3}$$

We claim that $\rho_{\delta V}(f_1, f_2) \equiv \rho_{V_\mu}(f_1, f_2)$ for

$$\mu(x, y) = \begin{cases} 1, & (x, y) \in \Delta ABC: \Delta ABC \in T, \Delta ABC \notin T_1 \text{ or} \\ & \Delta ABC \notin T_2; \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Indeed, this follows from definition of ρ_{V_μ} .

3 Methods

During merging process of two Delaunay triangulations T_1 and T_2 , some edges and triangles move to the united triangulation without changes and some of them are destroyed. So there are new edges and triangles, which connect nodes from different grids, in the general triangulation T .

We say that an edge or a triangle is called *interface* if it connects nodes from both of grids G_1, G_2 . On Fig. 2 all interface triangles constructed during merging are filled.

Measure (3) is calculated over interface triangles only. Naive algorithm to compute this measure examines each triangle and calculates difference volume (1) if this triangle is interface. For effective calculation of the measure more complex algorithm for extracting of interface triangles is needed.

Note also that in case when nodes of two grids are well mixed with one another all triangles of T can be interface. In this case values of measures (2) and (3) are equal. The advantage of measure (3) appears in cases when one surface is presented in more detail than the other. For example, in practise one can meet this case solving surface matching problem for surfaces acquired by two 3d scanners of different accuracy.

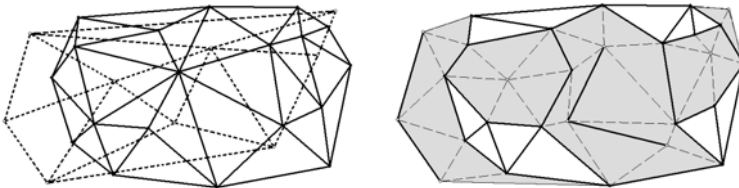


Fig. 2. Two triangulated grids (on the left) and united Delaunay triangulation with fill interface triangles (on the right)

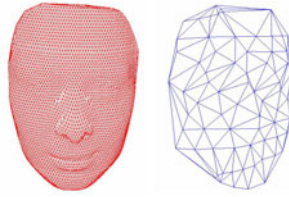


Fig. 3. Example of face models with different levels of detail

3.1 Algorithm for Interface Triangles Extraction

A set of interface triangles decomposes on several subsets. Each subset is a chain of triangles, which are pairwise incident by edges. This chain can be closed or open-ended (see Fig. 4). In the second case each of two terminal triangles has at least one boundary side, i.e., side from a convex hull $Conv(G)$ of G .

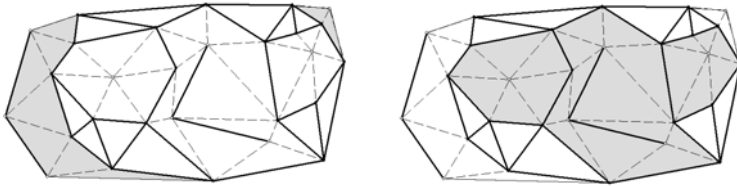


Fig. 4. Two open-ended chains (on the left) and three closed chains of interface triangles (on the right)

Therefore an algorithm for interface triangles extraction reduced to tracing of chains mentioned above.

Let us introduce concepts of connection for triangulation elements.

Definition 2. A set of nodes and edges (edges and triangles) of triangulation is called connected if for any pair of its nodes (edges) there exists a chain consisted of pairwise incident nodes and edges (edges and triangles).

Definition 3. A maximal connected set of nodes and edges is said to be a clout if it moves completely (without changes) to general triangulation T from one of the source triangulations.

Let us remark that the problem of chain tracing can be considered as the problem of clout location in triangulation.

We say that an interface edge of general triangulation T is called a starter if it belongs to a chain that is not traced yet. In other words, a starter initializes the process of chain tracing.

Hence the proposed algorithm for interface triangles extraction consists of the following stages:

1. Search for initial starter;
2. Tracing of chain conformable to the found starter;
3. Search for the next starter. If it exists move to the previous stage, otherwise finish.

Let us consider each stage in detail.

Search for First Starter. Assume that nodes of G_1 and G_2 are lexicographically ordered and $g_1[0], g_2[0]$ are the leftmost nodes of grids. Without loss of generality, we can assume that $g_1[0] \prec g_2[0]$.

Let us consider a circle that passes through $g_1[0]$ and $g_2[0]$ of center on horizontal ray with origin at $g_2[0]$ directed to the left.

Now we shall give the following definitions. A circle is called *empty* with respect to the grid G if it doesn't contain nodes of G . A circle is *incident to a point* if it passes through this point. A circle incident to a point is called a *maximal empty circle* if it is empty and has maximal radius. We say that an edge satisfies the *Delaunay condition* if there exists an empty circle that passes through the endpoints of this edge.

By construction, the considered circle is empty w.r.t. G_2 . At the same time it can contain some nodes $g_1[i_1], \dots, g_1[i_n]$ from G_1 . Let us construct a set of n different circles of center at the same horizontal ray that passes through one of these nodes. Choose a node $g_1[k] \in g_1[i_1], \dots, g_1[i_n]$ such that it conforms to the circle of minimal radius from the constructed set (see Fig. 5).

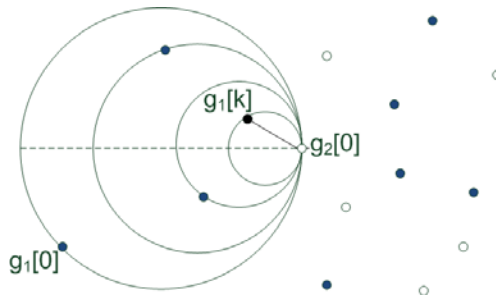


Fig. 5. Search for initial starter

The circle of minimal radius is empty w.r.t. G_1 and G_2 , consequently w.r.t. G . It follows that the edge connected nodes $g_1[k]$ and $g_2[0]$ satisfies the Delaunay condition and belongs to general triangulation T . The edge $g_1[k]g_2[0]$ is the *first starter* because the conditions of the following theorem hold.

Theorem 1. (*starter existence criterion*) *A node $g_1[i]$ of a grid G_1 has an incident interface edge of the general Delaunay triangulation constructed on $G_1 \cup G_2$ iff there exists a circle C such that the following conditions hold*

- (i) C is incident to this node;

- (ii) C is empty w.r.t. the set $G_1 \setminus \{g_1[i]\}$;
- (iii) C contains inside or on the boundary at least one node of the second grid G_2 .

Proof. Necessity. If two nodes of different grids G_1, G_2 form the interface edge of T then they have the incident empty circle that satisfies all required conditions by definition.

Sufficiency. Suppose $v \in G_1$ and there exists the circle such that it is incident to v , contains inside and/or on its boundary nodes $u_1, \dots, u_m \in G_2$ but is empty w.r.t. other nodes of G_1 . Let us consider a set of m embedded circles that pass through pairs of nodes $v, u_i, i = 1, \dots, m$ and have the common with the initial circle tangent at v . The circle from this set of minimal radius is incident to $v \in G_1, u_k \in G_2$ and empty w.r.t. $G_1 \cup G_2$. Therefore nodes v and u_k form the interface edge, which can be chosen as a starter.

Search time for the first starter is composed of time to compute the leftmost nodes of both grids and single look-up of nodes from G_2 during circle of minimal radius search. Thus time is linear in number of nodes in united grid.

Tracing of Chain of Interface Triangles. During tracing process sequential extraction of interface edges occurs. An interface edge extracted on a certain iteration is declared as *current*. The first starter becomes the first current edge.

Let AB be a current edge. During joining of edges AB and BA to lists of incident nodes for A and B , respectively, edges of these lists (bundles) that don't satisfy the Delaunay condition must be destroyed. Hence *bundle correction procedure* for interface edge is needed (see Fig. 6). It is based on checking the Delaunay condition by angle criterion [9].

The tracing process for chain using the found starter consists of the following steps:

- a) To declare the starter as a *current* edge;
- b) To correct bundles of endpoints of the current edge in the triangulations T_1, T_2 (destroy edges that don't satisfy the Delaunay condition);
- c) To construct a *new* edge that connects nodes of different grids;

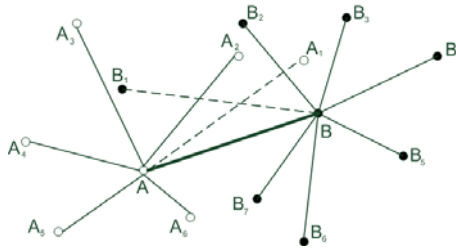


Fig. 6. Bundle correction for an interface edge AB . Edges AA_1 and BB_1 are destroyed

- d) To destroy edges in T_1 and T_2 that intersect the triangle formed by the current and new edges;
- e) To declare the new edge as a *current* and go to (b).

The tracing algorithm has analogy with an algorithm for Delaunay triangulation construction, which consists of cleaning and sewing together stages, from [10].

If on step (c) a new edge coincides with the starter then the traceable chain is closed. If we don't succeed in new edge construction then the current edge is a terminal edge of the chain. In the last case it is needed to continue the tracing process from the starter to the other side until the second terminal edge is found.

Search for Next Starter. Search of the rest of the starters is implemented using minimal spanning trees (MST) for Delaunay triangulations T_1, T_2 as if during chain tracing process connection of Delaunay triangulation is disturbed (see Definition 2) then there must be an edge of MST in the set of destroyed edges.

Let us introduce some definitions. We say that a *bridge* is an edge of MST for T_1 or T_2 such that it was destroyed during chain tracing process. An *influence circle of a bridge* is a circle such that this bridge is a diameter of it. One of the endpoints (nodes) of a bridge belongs to already traced chain and the other one is still a *free node*.

Lemma 1. *For any free node there exists an incident starter.*

In these terms, a free node, which has an incident starter for initialising chain tracing process, is generated during bridge destroying.

The following lemma is used to look for the second node of the starter:

Lemma 2. *Suppose B is a free node, A is a node of a triangulation T , a pair A, B forms a starter. Then B is inside of at least one of the maximal empty circles in T that are incident to A .*

Search for starter using the free node A and the bridge AB consists of the following steps:

- a) Search for traced interface triangle such that there is the free node A in the circumcircle of it (using search along AB);
- b) Search for interface triangles such that they are adjacent to the found triangle and there is A inside their circumcircles;
- c) To construct a set $D = \{D_1, \dots, D_n\}$ of nodes of the found triangles that are inside of the influence circle for the bridge AB ;
- d) To construct a circle of center on AB that passes through the nodes D_i, A for any $D_i \in D$ and select the node D_{i^*} that conforms to the circle of minimal radius (similarly to search for the first starter). The edge $D_{i^*}A$, which connects the selected node with the free one, is a new starter.

3.2 Computational Complexity of the Algorithm

In this subsection we present some auxiliary lemmas for evaluation of complexity of the proposed algorithm.

By the above a bridge is an edge of MST. MST of a triangulation is a subgraph of this triangulation [12]. This means that the influence circle of a bridge is empty w.r.t. endpoints and middle points of this bridge. Using this property we can estimate intersection measure between a bridge and the influence circle of another bridge.

Lemma 3. *A bridge can not cut from the influence circle of another bridge an arc of size more than 60° .*

Lemma 4. *Suppose AA_1 and BB_1 are bridges of a triangulation T_1 , both AA_1 and BB_1 intersect an edge PQ of a triangulation T_2 , node P is inside of both influence circles of the bridges AA_1 , BB_1 . Then different of angles $\angle APA_1$ and $\angle BPB_1$ is not less than 60° .*

Lemma 5. *Suppose an edge of T_1 intersects several bridges of T_2 . Then an endpoint of this edge is inside of influence circles of at the most two bridges.*

This implies that we can estimate number of bridges in T_1 that can be destroyed by an edge of T_2 .

Lemma 6. *During merging procedure of two Delaunay triangulations each edge intersects at the most four bridges that it destroys.*

The above four lemmas were proved by Mestetskiy, Tsarik in [13]. These lemmas are needed for proof of the following theorem.

Theorem 2. *Computational complexity of algorithm for interface triangles extraction is $O(N)$, where N is a number of nodes in general grid.*

The proof is omitted.

Let us consider main stages of general algorithm for computing measure (3) (see [14]) and complexity of each stage:

- 1) Construction of Delaunay triangulations T_1 , T_2 : $O(N_1 \log N_1) + O(N_2 \log N_2)$;
- 2) Construction of minimal spanning trees of triangulations using Cherion, Tarjan algorithm (see [15]): $O(N_1) + O(N_2)$;
- 3) Localization of each of two grids G_1 , G_2 in the triangulation of the other grid: $O(N)$ experimentally;
- 4) Interpolation each of two functions F_1 , F_2 on the grid that the other function is defined on: $O(N)$;
- 5) Extraction of all interface triangles: $O(N)$;
- 6) Summing V over all interface triangles, we obtain value of measure: $O(N)$.

Note that stage (5) doesn't use results of stages (3) and (4). It can be proved that stage (3) can be implemented during or after extraction procedure using its intermediate results. Hence theoretical complexity (not only experimental) of stage (3) is linear.

Therefore total computational complexity for measure computing of surfaces given by discrete models is $O(N \log N)$ because of triangulation construction stage. In case of surfaces are given by triangular models the complexity is $O(N)$.

4 Experiments

Computational experiments were carried out on 3d face models acquired by 3d scanner BroadwayTM designed by Artec Group Company [1]. The database consists of 48 models received by scanning of 8 different persons (6 different models for each person). Different models of one and the same person were used as surfaces for comparison.

Suppose S_1, S_2 are surfaces for comparison, S'_2 is a reduced (simplified) second surface. S'_2 is acquired from S_2 by uniform random thinning of the second grid. As the result of thinning 15% nodes of the second grid were removed.

Grids of S_1, S_2 are unstructured (nonuniform) but has approximately equal density. Hence we assume a value of measure (2) between them as adequate initial estimation. Grids of S_1 and S'_2 are unstructured grids with different density.

Table 1 shows an example of measure values for comparison of surfaces from the database. We see that the measure (3) estimates difference between surfaces S_1 and S'_2 more adequate than the measure (2). Values of both measures (2) and (3) are bigger than the initial estimation but the difference between (3) and the initial estimation is less. Similar results were received for the rest models of the database.

Table 1. Value of measures (2) & (3) for surface comparison

Measure	Value (mm)	Comments
$\rho_V(S_1, S_2)$	7094	Adequate estimation for S_1 & S_2
$\rho_V(S_1, S'_2)$	17963	$\rho_V(S_1, S'_2) > \rho_V(S_1, S_2)$
$\rho_{\delta V}(S_1, S'_2)$	10884	$\rho_{\delta V}(S_1, S'_2) \approx \rho_V(S_1, S'_2)$

5 Conclusions

New measure adapted for comparison problem of surfaces defined on unstructured grids with different density is introduced. An efficient algorithm for measure computing is proposed.

The measure allows only surface fragments that are represented by nodes of both grids. We call such fragments interface fragments. For efficiency of measure computing a new algorithm for interface triangles extraction is proposed. Computational complexity of the algorithm is presented.

Computing experiments for comparison of surfaces defined on grids with different density were carried out. As experimental estimations have shown, the introduced measure is adequate for such kind of source grids.

Acknowledgement. The author is grateful to professor Leonid Mestetskiy for his supervision, constant attention to this work and useful discussions. This research was partially supported by the Russian Foundation for Basic Research (grants 08-01-00670, 10-07-00609).

References

1. Artec Group — 3D Scanning Technologies, <http://www.artec-group.com>
2. Kolesov, A., Pavlova, O.: Surfer Package — Processing and Visualization of 2d Functions (in Russian). Computer Press (1999)
3. Dyshkant, N.: Disparity Measure Construction for Comparison of 3D Objects' Surfaces. In: Proceedings of the 2nd International Workshop on Image Mining, Theory and Applications (IMTA-2-2009), pp. 43–52. INSTICC Press, Lisbon (2009)
4. Dzaraev, C., Persin, L., Porochin, A.: Using of 3d Scanners for Diagnostics of Dentoalveolar Anomalies (in Russian). In: Theoretical and Practical Forum "Dental-Review", Moscow (2010)
5. Gruen, A., Akca, D.: Squares 3D Surface and Curve Matching. ISPRS Journal of Photogrammetry and Remote Sensing 59, 151–174 (2005)
6. Cignoni, P., Rocchini, C., Scopigno, R.: Metro: measuring error on simplified surfaces. In: Computer Graphics Forum, vol. 17, pp. 167–174. Blackwell Publishers, Malden (1998)
7. Fan, T., Medioni, G., Nevatia, R.: Recognizing 3D objects using surface descriptions. IEEE PAMI 11(11), 1140–1157 (1989)
8. Schenk, T.: Digital Photogrammetry. Terra-Science, Laurelville (1999)
9. Skvortsov, A.: A survey of algorithms for constructing a Delaunay triangulation (in Russian). In: Numerical Methods and Programming, vol. 3. RCC MSU Press (2002)
10. Mestetskiy, L.: Continuous Morphology of Binary Images: Figures, Skeletons and Circulars (in Russian). PHYSMATLIT Press, Moscow (2009)
11. Mestetskiy, L., Tsarik, E.: Delaunay triangulation: recursion without space division of vertices (in Russian). In: 14th International Conference on Computer Graphics and Vision GraphiCon 2004, pp. 267–270. CMC MSU Press, Moscow (2004)
12. Preparata, F., Shamos, M.: Computational Geometry: An Introduction. Springer, Heidelberg (1985)
13. Mestetskiy, L., Tsarik, E.: Merging of Unseparated Delaunay Triangulations. In: Complex Systems: Processing, Modelling and Optimization of Information, vol. 2, pp. 216–231. Tver State University, Tver (2004)
14. Dyshkant, N., Mestetskiy, L.: Comparison of One-Sheet Surfaces Acquired by 3D Scanning (in Russian). In: 18th International Conference on Computer Graphics and Vision GraphiCon 2008, pp. 270–277. CMC MSU Press, Moscow (2008)
15. Cheriton, D., Tarjan, R.: Finding minimum spanning trees. SIAM J. Comput. 5(4), 724–742 (1976)

Approximate Shortest Paths in Simple Polyhedra

Fajie Li¹ and Reinhard Klette²

¹ College of Computer Science and Technology
Huaqiao University, Xiamen, Fujian, China

² Computer Science Department, The University of Auckland
Private Bag 92019, Auckland 1142, New Zealand
li.fajie@hqu.edu.cn, r.klette@auckland.ac.nz

Abstract. Since the pioneering work by (Cohen and Kimmel, 1997) on finding a contour as a minimal path between two end points, shortest paths in volume images have raised interest in computer vision and image analysis. This paper considers the calculation of a Euclidean shortest path (ESP) in a three-dimensional (3D) polyhedral space Π . We propose an approximate $\kappa(\varepsilon) \cdot \mathcal{O}(M|V|)$ 3D ESP algorithm, not counting time for preprocessing. The preprocessing time complexity equals $\mathcal{O}(M|E| + |\mathcal{F}| + |V| \log |V|)$ for solving a special, but ‘fairly general’ case of the 3D ESP problem, where Π does not need to be convex. V and E are the sets of vertices and edges of Π , respectively, and \mathcal{F} is the set of faces (triangles) of Π . M is the maximal number of vertices of a so-called critical polygon, and $\kappa(\varepsilon) = (L_0 - L)/\varepsilon$ where L_0 is the length of an initial path and L is the true (i.e., optimum) path length. The given algorithm solves approximately three (previously known to be) NP-complete or NP-hard 3D ESP problems in time $\kappa(\varepsilon) \cdot \mathcal{O}(k)$, where k is the number of layers in a stack, which is introduced in this paper as being the *problem environment*. The proposed approximation method has straightforward applications for ESP problems when analyzing polyhedral objects (e.g., in 3D imaging), or for ‘flying’ over a polyhedral terrain.

1 Introduction and Related Work

Since the pioneering work in [10] on finding contours as minimal paths between two end points, minimal paths in volume images have raised interest in computer vision and image analysis (for example, [5, 11]). In medical image analysis, minimal paths were extracted in 3D images and applied to virtual endoscopy [1]. However, so far, minimal path computation is typically based on the *Fast Marching Method* which only considers grid points as the possible vertices of the minimal paths; but there exist Euclidean shortest paths such that *none* of its vertices is a grid point; see, e.g., the example in Section 4 of [17]. Thus, paths detected by the Fast Marching Method cannot be always the exact Euclidean shortest paths.

There already exist several approximation algorithms for 3D ESP calculations, and we briefly recall those. Pioneering the field, [21] presents an

$$\mathcal{O}(n^4(m + \log(n/\varepsilon))^2/\varepsilon^2)$$

algorithm for the general 3D ESP problem, where n is the descriptonal complexity of polyhedral scene elements (that is, vertices, edges, and faces of the polyhedron), ε the

accuracy of the algorithm, and m the maximum number of bits for representing a single integer coordinate of elements of the polyhedral scene. This was followed by [9], which presents an approximation algorithm for computing an $(1 + \varepsilon)$ -shortest path from p to q in time

$$\mathcal{O}(n^2 \lambda(n) \log(n/\varepsilon)/\varepsilon^4 + n^2 \log nr \log(n \log r))$$

where r is the ratio of the Euclidean distance $d_e(p, q)$ to the length of the longest edge of any given obstacle, and

$$\lambda(n) = \alpha(n)^{\mathcal{O}(\alpha(n)^{\mathcal{O}(1)})}$$

where $\alpha(n) = A^{-1}(n, n)$ is the inverse Ackermann function (see, e.g., [18]), which grows very slowly.

Assume a finite set of polyhedral obstacles in \mathbb{R}^3 . Let p, q be two points outside of the union of all obstacles, and $0 < \varepsilon < 1$. Reference [12] gives an $\mathcal{O}(\log(n/\varepsilon))$ algorithm to compute an $(1 + \varepsilon)$ -shortest path from p to q such that it does not intersect the interior of any obstacle. However, this algorithm requires a subdivision of \mathbb{R}^3 which may be computed in $\mathcal{O}(n^4/\varepsilon^6)$.

Given a convex partition of the free space, [2] presents an $\mathcal{O}((n/\varepsilon^3)(\log 1/\varepsilon)(\log n))$ algorithm for the general 3D ESP problem. Recently, [1] proposes algorithms for calculating approximate ESPs amid a set of convex obstacles. For latest results related to surface ESPs, see [4].

Altogether, the task of finding efficient and easy to implement solutions in this field is certainly challenging; see, for example, [19] saying on page 666 the following, when addressing mainly exact solutions: “The problem is difficult even in the most basic Euclidean shortest-path problem ... in a three-dimensional polyhedral domain Π , and even if the obstacles are convex, or the domain Π is simply connected.”

Rubberband algorithms provide a general strategy for solving ESP problems. See [16] for a general introduction into rubberband algorithms; they are characterized by sets of *steps*, defining possible locations of vertices of Euclidean shortest paths, a local optimization strategy, and a termination criterion; for example, see [16,22]. This class of algorithms emerged from a particular rubberband algorithm that was proposed in 2000 for calculating shortest paths in 3D image analysis [6,17].

In this paper, we apply a rubberband algorithm to present an approximate

$$\kappa(\varepsilon) \cdot \mathcal{O}(M|V|) + \mathcal{O}(M|E| + |S| + |V| \log |V|)$$

algorithm for ESP calculations when Π is a (type-2, see Definition 2 below) simply connected polyhedron which is not necessarily convex.

The given algorithm solves approximately three NP-complete or NP-hard 3D ESP problems in time $\kappa(\varepsilon) \cdot \mathcal{O}(k)$, where k is the number of layers in a stack, which is introduced as the *problem environment* below. Our algorithm has straightforward applications for ESP problems when analyzing polyhedral objects (e.g., in 3D imaging; for the extensive work using geodesics we just cite [26] as one example), or for ‘flying’ over a polyhedral terrain. The best known result for the latter problem is due to [27] by proposing an $\mathcal{O}((n/\varepsilon)(\log n)(\log \log n))$ algorithm for computing a $(2^{(p-1)/p} + \varepsilon)$ -approximation of an L_p -shortest path above a polyhedral terrain.

Section 2 provides necessary definitions and theorems. Section 3 describes our algorithm. Section 4 gives the time complexity of the algorithm. Section 5 illustrates the algorithm by some examples. Section 6 concludes the paper.

2 Basics

The algorithm in this paper generalizes a *rubberband algorithm* which was studied in [17] for computing ESPs in voxel-curves [14], where each voxel is a grid cube.

We denote by Π a *simple polyhedron* (i.e., a compact polyhedral region which is homeomorphic to a unit ball) in the 3D Euclidean space, which is equipped with an xyz Cartesian coordinate system. Let E be the set of edges of Π ; $V = \{v_1, v_2, \dots, v_n\}$ the set of vertices of Π . For $p \in \Pi$, let π_p be the plane which is incident with p and parallel to the xy -plane. The intersection $\pi_p \cap \Pi$ is a finite set of simple polygons; a singleton (i.e., a set only containing a single point) is considered to be a degenerate polygon.

Definition 1. A simple polygon P , being a connected component of $\pi_p \cap \Pi$, is called a *critical polygon* of Π (with respect to p).

Any vertex p defines in general a finite set of critical polygons. The notion of a critical polygon is also generalized as follows: We assume a simply connected (possibly unbounded) polyhedron Π , and we allow that the resulting (generalized) critical polygons may also be unbounded. For example, a generalized critical polygon may have a vertex at infinity, or it can be the complement of a critical polygon, as specified in Definition 1 (Section 5 will also make use of generalized critical polygons.)

Definition 2. We say that a simple polyhedron Π is a *type-1 polyhedron* iff any vertex p defines exactly one convex critical polygon. We say that a simple polyhedron Π is a *type-2 polyhedron* iff any vertex p defines exactly one simple critical polygon.

Obviously, each type-1 simple polyhedron is also a type-2 simple polyhedron. Our main algorithm below applies to type-2 simple polyhedra.

In what follows, Π is a type-2 simple polyhedron. For a simple polygon P , let P° be its topological interior, P^\bullet the closure of P° , and $\partial P = P^\bullet \setminus P^\circ$ the *frontier* of P . Let $\rho(p, q)$ be a path from p to q .

We recall some concepts introduced in [20]. Let (x_0, y_0, z_0) be a point in 3D space. Let

$$\begin{aligned} S_1 &= \{(x, y, z_0) : x_0 \leq x < \infty \wedge y_0 \leq y < \infty\} \\ S_2 &= \{(x, y, z_0) : -\infty < x \leq x_0 \wedge y_0 \leq y < \infty\} \\ S_3 &= \{(x, y, z_0) : -\infty < x \leq x_0 \wedge -\infty < y \leq y_0\} \\ S_4 &= \{(x, y, z_0) : x_0 \leq x < \infty \wedge -\infty < y \leq y_0\} \end{aligned}$$

S_i is called a *q-rectangle of type i* , where $i = 1, 2, 3, 4$. Furthermore, let (x_1, y_1, z_0) be a point in 3D space such that $x_1 > x_0$ and $y_1 > y_0$. Let

$$\begin{aligned} S_h &= \{(x, y, z_0) : -\infty < x < \infty \wedge y_0 \leq y \leq y_1\} \\ S_v &= \{(x, y, z_0) : x_0 \leq x \leq x_1 \wedge -\infty < y < \infty\} \end{aligned}$$

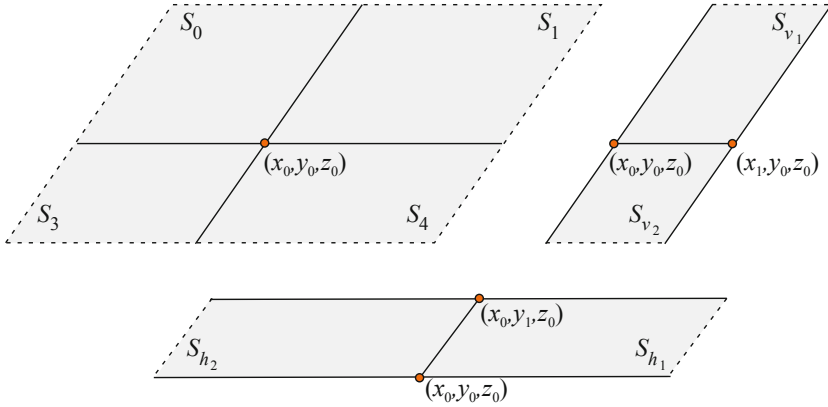


Fig. 1. Axis-aligned rectangles

Finally, let

$$\begin{aligned}
 S_{h_1} &= \{(x, y, z_0) : x_0 \leq x < \infty \wedge y_0 \leq y \leq y_1\} \\
 S_{h_2} &= \{(x, y, z_0) : -\infty < x \leq x_0 \wedge y_0 \leq y \leq y_1\} \\
 S_{v_1} &= \{(x, y, z_0) : x_0 \leq x \leq x_1 \wedge y_0 \leq y < \infty\} \\
 S_{v_2} &= \{(x, y, z_0) : x_0 \leq x \leq x_1 \wedge -\infty < y \leq y_0\}
 \end{aligned}$$

$S_h, S_v, S_{h_j},$ and S_{v_j} are called *horizontal* or *vertical strips*, for $j = 1, 2$. According to their geometric shape, we notice that

- (i) $S_1 [S_2, S_3, S_4]$ is unbounded in direction $(+x, +y) [(-x, +y), (-x, -y), (+x, -y)]$;
- (ii) $S_h [S_v]$ is unbounded in direction $\pm x [\pm y]$;
- (iii) $S_{h_1} [S_{h_2}, S_{v_1}, S_{v_2}]$ is unbounded in direction $+x [-x, +y, -y]$.

$S_i, S_h, S_v, S_{h_j},$ and S_{v_j} are also called *axis-aligned rectangles*, where $i = 1, 2, 3, 4$ and $j = 1, 2$. The stack S of axis-aligned rectangles is called *terrain-like* if, for at least one of the four directions $-x, +x, -y,$ or $+y$, each rectangle in S is unbounded (see Figure 1).

Let $\{s_1, s_2, \dots, s_m\}$ be a set of m line segments and S the union of those segments. Let p_1 and p_2 be two different points not in S ; see Fig. 2. We recall that points in \mathbb{R}^3 may be sorted by the lexicographic order of their coordinates.

Lemma 1. $d_e(p_1, p) + d_e(p_2, p) = \min\{d_e(p_1, q) + d_e(p_2, q) : q \in S\}$ and *lexicographic order* define a unique point in S , which can be computed in $\mathcal{O}(m)$ time.

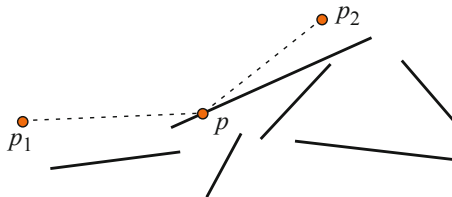


Fig. 2. Two points p_1 and p_2 and m segments

Proof. Consider $m = 1$. For line segment s_i , there is a unique point $q_i \in s_i$ such that

$$d_e(p_1, q_i) + d_e(p_2, q_i) = \min\{d_e(p_1, q) + d_e(p_2, q) : q \in s_i\}$$

Consider case $m = 2$ and points q_1 and q_2 . If $d_e(p_1, q_1) + d_e(p_2, q_1) < d_e(p_1, q_2) + d_e(p_2, q_2)$, then $p = q_1$. If $d_e(p_1, q_1) + d_e(p_2, q_1) > d_e(p_1, q_2) + d_e(p_2, q_2)$, then $p = q_2$. If $d_e(p_1, q_1) + d_e(p_2, q_1) = d_e(p_1, q_2) + d_e(p_2, q_2)$, then we decide for that point which comes first in lexicographic order. Cases $m > 2$ follow analogously. \square

Let P be a convex critical polygon of Π , defined by the plane $z = c$. Let p_1 and p_2 be two points such that their z -coordinates satisfy $z_1 < c < z_2$. Then we also have the following (see Theorem 23, page 146, [15]):

Lemma 2. *There is a unique point $p \in P^\bullet$ such that*

$$d_e(p_1, p) + d_e(p_2, p) = \min\{d_e(p_1, q) + d_e(p_2, q) : q \in P^\bullet\}$$

3 ESP Computation

We start by presenting a procedure, used by a rubberband algorithm (Algorithm 1 below), and then frequently called in the main algorithm (Algorithm 3) of this paper.

Let $\mathcal{F} = \{F_1, F_2, \dots, F_m\}$ be the set of all faces of Π , and V the set of all vertices of Π . – The following very basic Procedure 1 simply ‘walks around’ the polyhedron by tracing an intersection with a given plane. We do not detail this procedure; it is a fairly straightforward isoheight trace of a polyhedron, assuming that the data structure of the polyhedron links edges to faces.

Procedure 1. (compute a sequence of vertices of the critical polygon; see Fig. 3)

Input: Set \mathcal{F} and a vertex $v \in V$ such that π_v intersects Π in more than just one point.

Output: An ordered sequence of all vertices in V_v , which is the vertex set of the critical polygon P_v .

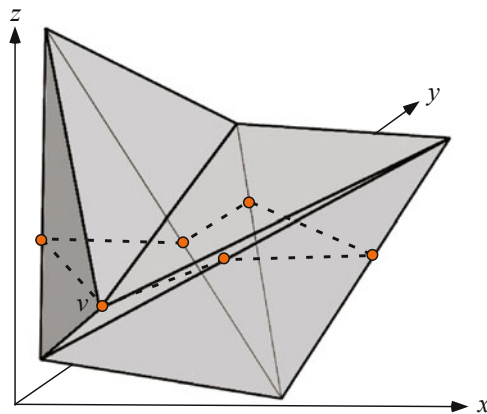


Fig. 3. The labeled vertex v identifies a sequence of six vertices of the critical polygon P_v , defined by the intersection of plane π_v with the shown (Schönhardt) polyhedron

The main ideas of the rubberband algorithm below (Algorithm 1) are as follows: For a start, we randomly take a point in the closure of each critical polygon to identify an initial path from p to q . Then we enter a loop; in each iteration, we optimize locally the position of point p_1 by moving it within its critical polygon, then of p_2, \dots , and finally of p_k . At the end of each iteration, we check the difference between the length of the current path to that of the previous one; if it is less than a given accuracy threshold $\varepsilon > 0$ then we stop. Otherwise, we go to the next iteration. - Let p_x be the x -coordinate of point p , v_{1z} the z -coordinate of point v_1 , and so forth.

Algorithm 1. (a rubberband algorithm for type-1 polyhedra)

Input: Two points p and q , a set $\{P_{v_1}^\bullet, P_{v_2}^\bullet, \dots, P_{v_k}^\bullet\}$, where P_{v_i} is a critical polygon of a given polyhedron Π , k vertices $v_i \in \partial P_{v_i}$ such that $p_z < v_{1z} < \dots < v_{kz} < q_z$, for $i = 1, 2, \dots, k$, and there is no any other critical polygon of Π between p and q ; given is also an accuracy constant $\varepsilon > 0$.

Output: The set of all vertices of an approximate shortest path which starts at p , then visits approximate optimal positions p_1, p_2, \dots, p_k in that order, and finally ends at q .

- 1: For each $i \in \{1, 2, \dots, k\}$, let the initial vertex p_i be a vertex of $P_{v_i}^\bullet$.
- 2: Let $L_0 = \infty$. Calculate $L_1 = \sum_{i=0}^k d_e(p_i, p_{i+1})$, where $p_0 = p$ and $p_{k+1} = q$.
- 3: **while** $L_0 - L_1 \geq \varepsilon$ **do**
- 4: **for** $i = 1, 2, \dots, k$ **do**
- 5: Compute a point $q_i \in P_{v_i}^\bullet$ such that

$$d_e(p_{i-1}, q_i) + d_e(q_i, p_{i+1}) = \min\{d_e(p_{i-1}, p) + d_e(p, p_{i+1}) : p \in P_{v_i}^\bullet\}$$
- 6: Update the path $\langle p, p_1, p_2, \dots, p_k, q \rangle$ by replacing p_i by q_i .
- 7: **end for**
- 8: Let $L_0 = L_1$ and calculate $L_1 = \sum_{i=0}^k d_e(p_i, p_{i+1})$.
- 9: **end while**
- 10: Return $\langle p, p_1, p_2, \dots, p_k, q \rangle$.

The set $\{P_{v_1}^\bullet, P_{v_2}^\bullet, \dots, P_{v_k}^\bullet\}$ in Algorithm 1 is called the *step set* of this rubberband algorithm. (Identifying a ‘suitable’ step set is normally the main issue when defining a rubberband algorithm; see, for example, discussion of step sets in [16,22].)

Theorem 1. *If Π is a type-1 polyhedron in the input of Algorithm 1 then the solution obtained by Algorithm 1 is an approximate global solution to the 3D ESP problem.*

Proof. Let $X = \Pi_{i=1}^k P_{u_i}^\bullet \subset \mathbb{R}^k$, where $P_{u_i}^\bullet$ is as defined in Algorithm 1. As Π is a type-1 polyhedron, then P_{u_i} is a convex polygon, where $i = 1, 2, \dots, k$. Let $Y \subset X$ be the set of all solutions obtained by Algorithm 1 for any initialization in X and the given $\varepsilon > 0$.

As each P_{u_i} is a convex polygon, by Lemma 2, the point q_i in Step 5 of Algorithm 1 is unique, and q_i depends continuously upon p_{i-1} and p_{i+1} defined in Step 5 of Algorithm 1. Thus, Algorithm 1 defines a continuous function, denoted by f , mapping from X (i.e., an initialization) into Y , with values depending on the used accuracy $\varepsilon > 0$.²

¹ If $p_{i-1}p_{i+1} \cap P_{v_i}^\circ \neq \emptyset$, then assign $q_i \leftarrow p_{i-1}p_{i+1} \cap P_{v_i}^\circ$.

² If each $P_{u_i}^\bullet$ is degenerated into a single edge, then there exists a unique solution to the ESP problem; independent of the chosen initialization, solutions will converge to this unique solution if ε goes to zero; see [8,23,25].

Now let $\bar{v} = (v_1, v_2, \dots, v_k) \in Y$. Then v_i is either located on an edge of polygon P_{u_i} , which is contained in the frontier ∂P_{u_i} , for $i = 1, 2, \dots, k$, or v_i is located in the interior $P_{u_i}^\circ$, and v_{i-1}, v_i and v_{i+1} are collinear. Thus, Y is a finite set.

It remains to prove that Y is a singleton. Let $\bar{v}_0 \in Y$; we have that $f^{-1}(\bar{v}_0) \subset X$. For each initialization $\bar{v} \in f^{-1}(\bar{v}_0)$, as f is a continuous function, there exists a sufficiently small open neighborhood (with respect to the usual topology on \mathbb{R}^k) of \bar{v} , denoted by $N(\bar{v}, \delta_{\bar{v}})$, such that for each $\bar{v}' \in N(\bar{v}, \delta_{\bar{v}})$, $f(\bar{v}') = \bar{v}_0$. Thus, $N(\bar{v}, \delta_{\bar{v}}) \subseteq f^{-1}(\bar{v}_0)$ and $\cup_{\bar{v} \in f^{-1}(\bar{v}_0)} N(\bar{v}, \delta_{\bar{v}}) \subseteq f^{-1}(\bar{v}_0)$.

On the other hand, because (simply by definition) $f^{-1}(\bar{v}_0) = \{\bar{v} : \bar{v}_0 = f(\bar{v})\}$ and $\bar{v} \in N(\bar{v}, \delta_{\bar{v}})$, we have that $f^{-1}(\bar{v}_0) \subseteq \cup_{\bar{v} \in f^{-1}(\bar{v}_0)} N(\bar{v}, \delta_{\bar{v}})$. Therefore, $f^{-1}(\bar{v}_0) = \cup_{\bar{v} \in f^{-1}(\bar{v}_0)} N(\bar{v}, \delta_{\bar{v}})$. Because $N(\bar{v}, \delta_{\bar{v}})$ is an open set, $f^{-1}(\bar{v}_0)$ is also open. Let

$$f^{-1}(\bar{v}_0) = \cup_{i=1}^k S_i$$

where S_i is an open subset of $P_{u_i}^\bullet$, for $i = 1, 2, \dots, k$. Recall that $f^{-1}(\bar{v}_0) \subset X$, thus there exists an S_i such that $\emptyset \subset S_i \subset P_{u_i}^\bullet$.

Without loss of generality, suppose that $\emptyset \subset S_1 \subset P_{u_1}^\bullet$. This implies that there exists a point $(x_0, y_0) \in P_{u_1}^\bullet$ such that $\emptyset \subset S_1|_{x_0} \subset P_{u_1}^\bullet|_{x_0}$ [24]. Thus, $S_1|_{x_0}$ is a nonempty open subset of $P_{u_1}^\bullet|_{x_0}$. Set $S_1|_{x_0}$ is a union of a countable number of open or half-open intervals (see Proposition 5.1.4 in [24]).

Thus, there exists a point $w_1 \in P_{u_1}^\bullet|_{x_0} \setminus S_1$ such that, for every positive ε_1 , there exists a point $w'_1 \in N(w_1, \varepsilon_1) \cap S_1$ [again, $N(w_1, \varepsilon_1)$ is an open neighborhood with respect to the usual topology on $P_{u_1}^\bullet$]. Therefore, there exists a point $\bar{v}_1 \in X \setminus f^{-1}(\bar{v}_0)$ such that, for each positive ε_1 , there exists a point $\bar{v}'_1 \in N(\bar{v}_1, \varepsilon_1) \cap f^{-1}(\bar{v}_0)$. This contradicts that f is a continuous function on X . Thus, Y is a singleton. \square

As an informal interpretation of this proof: If ε is sufficiently small then Y is a neighborhood $N(\bar{v}, \delta_{\bar{v}})$ of a single point \bar{v} , where $\delta_{\bar{v}}$ is sufficiently small (depending on ε) such that computers regard $N(\bar{v}, \delta_{\bar{v}})$ as a single point \bar{v} because of rounding.

If input Π is a type-2 polyhedron then the solution obtained by Algorithm 1 might not be an approximate global solution to the 3D ESP problem. However, following Theorem 1 we propose the following modification of Algorithm 1 for type-2 polyhedrons, with an initial mapping of non-convex polygons on their convex hulls:

Algorithm 2. (a rubberband algorithm for type-2 polyhedra)

Both input and output are the same as in Algorithm 1.

- 1: For $i \in \{1, 2, \dots, k\}$, apply (e.g.) the Melkman algorithm for computing $C(P_{v_i})$, the convex hull of P_{v_i} .
- 2: Let $C(P_{v_1}^\bullet), C(P_{v_2}^\bullet), \dots, C(P_{v_k}^\bullet), p$, and q be the input of Algorithm 1 for computing an approximate shortest route $\langle p, p_1, \dots, p_k, q \rangle$.
- 3: For $i = 1, 2, \dots, k - 1$, find³ a point $q_i \in C(P_{v_i}^\bullet)$ such that $d_e(p_{i-1}, q_i) + d_e(q_i, p_{i+1}) = \min\{d_e(p_{i-1}, p) + d_e(p, p_{i+1}) : p \in C(P_{v_i}^\bullet)\}$. Update the path for each i by $p_i = q_i$.

³ $S|_{x_0} = \{(x_0, y) : (x, y) \in S \wedge x = x_0\}$

⁴ If $p_{i-1}p_{i+1} \cap P_{v_i}^\circ \neq \emptyset$, then set $q_i \leftarrow p_{i-1}p_{i+1} \cap P_{v_i}^\circ$.

- 4: Let $P_{v_1}^\bullet, P_{v_2}^\bullet, \dots, P_{v_k}^\bullet, p$ and q be the input of Algorithm 1 and points p_i as obtained in Step 3 are the initial vertices p_i in Step 1 of Algorithm 1. Continue with running Algorithm 1.
- 5: Return $\langle p, p_1, \dots, p_{k-1}, p_k, q \rangle$ as provided in Step 4.

Step 2 iterates through the closures of convex hulls. The iteration through step sets $C(P_{v_i}^\bullet)$ only occurs in Step 4 (i.e., when applying Subalgorithm 1 for a second time, using the same ε). Algorithm 2 provides an $(1 + (L_2 - L_1)/L)$ -approximate global solution for the ESP, where L is the length of an optimal path; L_1 is the length of the path obtained in Step 2; L_2 the length of the final path obtained in Step 5. Note that $L_2 \geq L_1$, and $L_2 = L_1$ if all polygons P_i are convex. Also note that $L < L_1$, then $(1 + (L_2 - L_1)/L) \leq L_2/L_1$. Thus, Algorithm 2 provides an L_2/L_1 -approximate global solution for the ESP problem.

The main ideas of Algorithm 3 below are as follows: We apply Procedure 1 to compute the step set of a rubberband algorithm as given in Algorithms 1 or 2. Then we simply apply this rubberband algorithm to compute (of course, approximately only, defined by the chosen accuracy ε) the ESP.

For the input polyhedron we assume that it is of type-2. For example, the Schönhardt polyhedron as shown in Fig. 3 is of type-2, but it might be rotated such that the resulting polyhedron is not of type-2 anymore.

Algorithm 3. (main algorithm)

Input: Two points p and q in Π ; sets \mathcal{F} and V of faces and vertices of Π , respectively.

Output: The set of all vertices of an approximate shortest path, starting at p and ending at q , and contained in Π .

- 1: Initialize $V' \leftarrow \{v : p_z < v_z < q_z \wedge v \in V\}$.
- 2: Sort V' according to the z -coordinate.
- 3: We obtain $V' = \{v_1, v_2, \dots, v_{k'}\}$ with $v_{1z} \leq v_{2z} \leq \dots \leq v_{k'z}$.
- 4: Partition V' into pairwise disjoint subsets V_1, V_2, \dots , and V_k such that $V_i = \{v_{i1}, v_{i2}, \dots, v_{in_i}\}$, with $v_{ijz} = v_{i,j+1z}$, for $j = 1, 2, \dots, n_i - 1$, and $v_{i1z} < v_{i+1,1z}$, for $i = 1, 2, \dots, k - 1$. [That is, this step partitions the set V' into some subsets such that the points in the same subset have an identical z -coordinate.]
- 5: Set $u_i \leftarrow v_{i1}$, where $i = 1, 2, \dots, k$.
- 6: Set $V'' \leftarrow \{u_1, u_2, \dots, u_k\}$ (then we have that $u_{1z} < u_{2z} < \dots < u_{kz}$).
- 7: **for** each $u_i \in V''$ **do**
- 8: Apply Procedure 1 for computing V_{u_i} (i.e., a sequence of vertices of the critical polygon P_{u_i}).
- 9: **end for**
- 10: Set $\mathcal{F}_{step} \leftarrow \{P_{u_1}^\bullet, P_{u_2}^\bullet, \dots, P_{u_k}^\bullet\}$.
- 11: Set $P \leftarrow \{p\} \cup V'' \cup \{q\}$.
- 12: Apply Algorithm 2 on inputs \mathcal{F}_{step} and P , for computing the shortest path $\rho(p, q)$ inside of Π .
- 13: Convert $\rho(p, q)$ into the standard form of a shortest path by deleting all vertices which are not on any edge of Π (i.e., delete p_i if p_i is not on an edge of P_{u_i}).

By the discussions after Algorithm 2, we have the following

Theorem 2. Algorithm 3 provides an L_2/L_1 -approximate global solution for the ESP problem, where L_1 is the length of the path obtained in Step 2 of Algorithm 2. L_2 is the length of the final path obtained in Step 5 of Algorithm 2.

4 Time Complexity

At first, we can show (calculation of upper time bounds for involved steps is fairly straightforward) that Procedure 1 can be computed in $\mathcal{O}(|V_v||E(S_v)|+|\mathcal{F}|)$ time, where $S_v = \{F : F \in \mathcal{F} \wedge e = uv \in E(F) \wedge (u_z \leq v_z \leq w_z \vee w_z \leq v_z \leq u_z)\}$. Then we can show that the time complexity of Algorithm 1 equals $\kappa(\varepsilon) \cdot \mathcal{O}(\sum_{j=1}^k |V_{v_j}|)$, where $\kappa(\varepsilon)$ is the number of iterations of the while loop in Algorithm 1. By Lemma 1, Step 5 can be computed in $\mathcal{O}(|V_{v_j}|)$ time, where V_{v_j} is as in Algorithm 1 for $j = 1, 2, \dots, k$. Thus, each iteration of Algorithm 1 can be computed in $\mathcal{O}(\sum_{j=1}^k |V_{v_j}|)$ time. Obviously, Algorithm 2 has the same time complexity as Algorithm 1. These three results allow us then to show that Algorithm 3 can be computed in

$$\kappa(\varepsilon) \cdot \mathcal{O}\left(\sum_{j=1}^k |V_{u_j}|\right) + \mathcal{O}\left(\sum_{j=1}^k |V_{u_j}||E(S_{u_j})| + |\mathcal{F}| + |V| \log |V|\right)$$

where the second term is the time for preprocessing. This can finally be reformulated in the more compact form that Algorithm 3 is of complexity

$$\kappa(\varepsilon) \cdot \mathcal{O}(M|V|) + \mathcal{O}(M|E| + |\mathcal{F}| + |V| \log |V|)$$

for $M = \max\{|V_{u_j}| : j = 1, 2, \dots, k\}$, where the second $\mathcal{O}(\dots)$ term is the time for preprocessing.

In Algorithm 1 let $\kappa(\varepsilon) = \frac{L_0-L}{\varepsilon}$ be a function which only depends upon the difference between the lengths L_0 of an initial path and L of the optimum path, and the accuracy constant ε . Let L_m be the length of the m -th updated path, for $m = 0, 1, 2, \dots$, with $L_m - L_{m+1} \geq \varepsilon$ (otherwise the algorithm stops). It follows that

$$\kappa(\varepsilon) = \frac{L_0 - L}{\varepsilon} \geq 1 + \frac{L_1 - L}{\varepsilon} \geq \dots \geq m + \frac{L_m - L}{\varepsilon}$$

The sequence $\{m + \frac{L_m-L}{\varepsilon}\}$ is monotonously decreasing, lower bounded by 0, and stops at the first m_0 where $L_{m_0} - L_{m_0+1} < \varepsilon$.

We have implemented a simplified version of Algorithm 1 where all $P_{v_i}^\bullet$ s were de-generated to be line segments. Thousands of experimental results indicated that $\kappa(\varepsilon)$ does not depend on the number k of segments but the value of ε . We selected $\varepsilon = 10^{-15}$ and k was in between 4 and 20,000, the observed maximal value of $\kappa(\varepsilon)$ was 380,000. It shows that the smallest upper bound of $\kappa(\varepsilon) \geq \kappa(10^{-15}) \geq 380,000$. In other words, the number of iterations in the while-loop can be huge even for some small value of k . On the other hand, all these experimental results indicated that $|L_m - L_{m+1}| \leq 1.2$, when $m > 200$ and L was between 10,000 and 2,000,000. It showed that $\kappa(1.2) \leq 200$ and the relative error $|L_m - L_{m+1}|/L \leq 1.2 \times 10^{-4}$. In other words, these experiments showed that the algorithm already reached an approximate ESP with a very minor relative error after 200 iterations of the while loop; the remaining iterations were ‘just’ spent on improving a very small fraction of the length of the path.

5 Examples: Three NP-Complete or NP-Hard Problems

We apply Algorithms 2 and 3 for approximate solutions of hard problems, characterized below (by appropriate references) as being NP-complete or NP-hard. Let $p, q \in \Pi$ such that $p_z < q_z$. Let $V_{pq} = \{v : p_z < v_z < q_z \wedge v \in V\}$, where V is the set of all vertices of Π . For doing so, we are allowing for input polyhedra different from the bounded type-2 polyhedra so far, but only input polyhedra which allow us to use those algorithms without any further modification.

We consider unbounded polyhedra (which also satisfy the type-2 constraint), and, thus, generalized critical polygons.

Example 1. Let Π be a simply connected polyhedron such that each critical polygon is the complement of an axis-aligned rectangle. Following Section 4, the Euclidean shortest path between p and q inside of Π can be approximately computed in $\kappa(\varepsilon) \cdot \mathcal{O}(|V_{pq}|)$ time. Therefore, the 3D ESP problem can be approximately solved efficiently in such a special case. Finding the exact solution is NP-complete because of the following

Theorem 3. ([20], Theorem 4) *It is NP-complete to decide whether there exists an obstacle-avoiding path of Euclidean length at most L among a set of stacked axis-aligned rectangles (see Fig. 4). The problem is (already) NP-complete for the special case that the axis-aligned rectangles are all q -rectangles of types 1 or 3.*

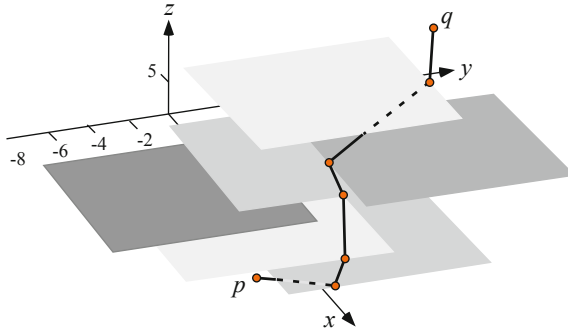


Fig. 4. A path from p to q which does not intersect any of the shown rectangles at an inner point

Example 2. Let Π be a simply connected polyhedron such that each critical polygon is the complement of a triangle. Following Section 4, the Euclidean shortest path between p and q inside of Π can be approximately computed in $\kappa(\varepsilon) \cdot \mathcal{O}(|V_{pq}|)$ time. Finding the exact solution is NP-hard because of the following

Theorem 4. ([7]) *It is NP-hard to decide whether there exists an obstacle-avoiding path of Euclidean length at most L among a set of stacked triangles.*

Example 3. Let S be a stack of k horizontal or vertical strips. The Euclidean shortest path among S can be approximately computed in $\kappa(\varepsilon) \cdot \mathcal{O}(k)$ time. Finding the exact solution is NP-complete because of the following

Theorem 5. ([20], Theorem 5) *It is NP-complete to decide whether there exists an obstacle-avoiding path of Euclidean length at most L among a finite number of stacked horizontal and vertical strips.*

Example 4. Let \mathcal{S} be a stack of k terrain-like axis-parallel rectangles. The Euclidean shortest path among \mathcal{S} can be approximately computed in $\kappa(\varepsilon) \cdot \mathcal{O}(k)$ time. The best known algorithm for finding the exact solution has a time complexity in $\mathcal{O}(k^4)$ due to the following

Theorem 6. ([20], Theorem 6) *Let \mathcal{S} be a stack of k terrain-like axis-parallel rectangles. The Euclidean shortest path among \mathcal{S} can be computed in $\mathcal{O}(k^4)$ time.*

6 Conclusions

This paper described an algorithm for solving the 3D ESP problem when the domain Π is a type-2 simply connected polyhedron; the algorithm has a time complexity in $\kappa(\varepsilon) \cdot \mathcal{O}(M|V|) + \mathcal{O}(M|E| + |\mathcal{F}| + |V| \log |V|)$ (where $\mathcal{O}(M|E| + |\mathcal{F}| + |V| \log |V|)$ is the time for preprocessing). It was also shown that the algorithm approximately solves three NP-complete or NP-hard problems in time $\kappa(\varepsilon) \cdot \mathcal{O}(k)$, where k is the number of layers in the given stack of polygons.

Our algorithm has straightforward applications on ESP problems in 3D imaging (where proposed solutions depend on geodesics), or when ‘flying’ over a polyhedral terrain. The best result so far for the latter problem was an $\mathcal{O}((n/\varepsilon)(\log n)(\log \log n))$ algorithm for computing a $(2^{(p-1)/p} + \varepsilon)$ -approximation to the L_p -shortest path above a polyhedral terrain.

As there does not exist an algorithm for finding exact solutions to the general 3D ESP problem (see Theorem 9, [3]), our method defines a new opportunity to find approximate (and efficient!) solutions to the discussed classical, fundamental, hard and general problems.

References

1. Agarwal, P.K., Sharathkumar, R., Yu, H.: Approximate Euclidean shortest paths amid convex obstacles. In: Proc. ACM-SIAM Sympos. Discrete Algorithms, pp. 283–292 (2009)
2. Aleksandrov, L., Maheshwari, A., Sack, J.-R.: Approximation algorithms for geometric shortest path problems. In: Proc. ACM Sympos. Theory Comput., pp. 286–295 (2000)
3. Bajaj, C.: The algebraic complexity of shortest paths in polyhedral spaces. In: Proc. Allerton Conf. Commun. Control Comput., pp. 510–517 (1985)
4. Balasubramanian, M., Polimeni, J.R., Schwartz, E.L.: Exact geodesics and shortest paths on polyhedral surfaces. IEEE Trans. Pattern Analysis Machine Intelligence 31, 1006–1016 (2009)
5. Benmansour, F., Cohen, L.D.: Fast object segmentation by growing minimal paths from a single point on 2D or 3D images. J. Math. Imaging Vision 33, 209–221 (2009)
6. Buelow, T., Klette, R.: Rubber band algorithm for estimating the length of digitized space-curves. In: Proc. ICPR, vol. III, pp. 551–555. IEEE, Los Alamitos (2000)
7. Canny, J., Reif, J.H.: New lower bound techniques for robot motion planning problems. In: Proc. IEEE Conf. Foundations Computer Science, pp. 49–60 (1987)

8. Choi, J., Sellen, J., Yap, C.-K.: Precision-sensitive Euclidean shortest path in 3-space. In: Proc. ACM Sympos. Computational Geometry, pp. 350–359 (1995)
9. Clarkson, K.L.: Approximation algorithms for shortest path motion planning. In: Proc. ACM Sympos. Theory Comput., pp. 56–65 (1987)
10. Cohen, L.D., Kimmel, R.: Global minimum for active contour models: a minimal path approach. *Int. J. Computer Vision* 24, 57–78 (1997)
11. Deschamps, T., Cohen, L.D.: Fast extraction of minimal paths in 3D images and applications to virtual endoscopy. *Med. Image Anal.* 5, 281–299 (2001)
12. Har-Peled, S.: Constructing approximate shortest path maps in three dimensions. In: Proc. ACM Sympos. Computational Geometry, pp. 125–130 (1998)
13. Klette, R., Rosenfeld, A.: *Digital Geometry*. Morgan Kaufmann, San Francisco (2004)
14. Li, F., Klette, R.: The Class of Simple Cube-Curves Whose MLPs Cannot Have Vertices at Grid Points. In: Andrès, É., Damiani, G., Lienhardt, P. (eds.) *DGCI 2005*. LNCS, vol. 3429, pp. 183–194. Springer, Heidelberg (2005)
15. Li, F., Klette, R.: Exact and approximate algorithms for the calculation of shortest paths. IMA Minneapolis, Report 2141(2006), www.ima.umn.edu/preprints/oct2006
16. Li, F., Klette, R.: Rubberband algorithms for solving various 2D or 3D shortest path problems. In: Proc. Computing: Theory and Applications, The Indian Statistical Institute, Kolkata, pp. 9–18. IEEE, Los Alamitos (2007)
17. Li, F., Klette, R.: Analysis of the rubberband algorithm. *Image Vision Computing* 25, 1588–1598 (2007)
18. Liu, Y.A., Stoller, S.D.: Optimizing Ackermann’s function by incrementalization. In: Proc. ACM SIGPLAN Sympos. Partial Evaluation Semantics-Based Program Manipulation, pp. 85–91 (2003)
19. Mitchell, J.S.B.: Geometric shortest paths and network optimization. In: Sack, J.-R., Urrutia, J. (eds.) *Handbook of Computational Geometry*, pp. 633–701. Elsevier, Amsterdam (2000)
20. Mitchell, J.S.B., Sharir, M.: New results on shortest paths in three dimensions. In: Proc. ACM Sympos. Computational Geometry, pp. 124–133 (2004)
21. Papadimitriou, C.H.: An algorithm for shortest path motion in three dimensions. *Inform. Processing Letters* 20, 259–263 (1985)
22. Pan, X., Li, F., Klette, R.: Approximate shortest path algorithms for sequences of pairwise disjoint simple polygons. In: Proc. Canadian Conf. Computational Geometry, Winnipeg, Canada, pp. 1–4 (2010)
23. Sharir, M., Schorr, A.: On shortest paths in polyhedral spaces. *SIAM J. Comput.* 15, 193–215 (1986)
24. Wachsmuth, B.G.: Interactive real analysis, <http://web01.shu.edu/projects/reals/topo/index.html> (last visit: May, 2009)
25. Yap, C.-K.: Towards exact geometric computation. *Computational Geometry: Theory Applications* 7, 3–23 (1997)
26. Wang, Y., Peterson, B.S., Staib, L.H.: 3D brain surface matching based on geodesics and local geometry. *Computer Vision Image Understanding* 89, 252–271 (2003)
27. Zadeh, H.Z.: Flying over a polyhedral terrain. *Inform. Processing Letters* 105, 103–107 (2008)

Author Index

- Aiger, Dror 223
Andres, Eric 235, 296, 358
Arlicot, Aurore 199
- Batenburg, K. Joost 369
Berthé, Valérie 47
Bertrand, Gilles 129
Bhowmick, Partha 489
Biswas, Arindam 489
Bloch, Isabelle 429
Blondin Massé, Alexandre 381
Bretto, Alain 429
Brunetti, Sara 394
Buzer, Lilian 223
- Cartade, Colin 59
Chandra, Shekhar 406
Ciril, Igor 465
Čomić, Lidija 477
Couprie, Michel 141, 163, 175
Cousty, Jean 441
- Daněk, Ondřej 71
Darbon, Jérôme 465
De Florian, Leila 477
Desolneux, Agnès 1
Dias, Fábio 441
Dulio, Paolo 394
Dutt, Mousumi 489
Dyshkant, Natalia 501
- Evenou, Pierre 199
- Feschet, Fabien 83
Fortes, Wagner 369
Frosini, Andrea 381
Fuchs, Laurent 296
- Gérard, Yan 83, 284
Gonzalez-Diaz, Rocio 153
Gupta, Raj Kumar 272
- Hajdu, Lajos 369
- Jacob-Da Col, Marie-Andrée 187
Jimenez, Maria-Jose 153
- Kenmochi, Yukiko 223
Kerautret, Bertrand 247
Klette, Gisela 260
Klette, Reinhard 513
- Labbé, Sébastien 47
Lachaud, Jacques-Olivier 247, 308, 320
Largeteau-Skapin, Gaëlle 296, 358
Leung, Maylor K.H. 272
Li, Fajie 513
- Malgouyres, Rémy 59
Matula, Pavel 71
Mazo, Loïc 163
Medrano, Belen 153
Mercat, Christian 59
Mesmoudi, Mohammed Mostefa 477
Monteil, Thierry 95
- Naegel, Benoît 453
Najman, Laurent 441
Nguyen, Thanh Phuong 247
Normand, Nicolas 199, 417
- Ouattara, Jean-Serge Dimitri 296
- Pacheco, Ana 104
Passat, Nicolas 163, 453
Peri, Carla 394
Prasad, Dilip K. 272
Provot, Laurent 83, 284
- Raynal, Benjamin 175
Real, Pedro 104
Rebatel, Fabien 116
Richard, Aurélie 296
Rinaldi, Simone 381
Rodríguez, Marc 296
Ronse, Christian 163
Rossignac, Jarek 13
Roussillon, Tristan 235, 308
- Said, Mouhammad 320
Samir, Chafik 59

Serra, Jean 35
Sivignon, Isabelle 333
Strand, Robin 199, 211
Svalbe, Imants 406, 417

Talbot, Hugues 223
Tellier, Pierre 187

Thiel, Édouard 116
Tijdeman, Robert 369

Veelaert, Peter 346
Vuillon, Laurent 381

Zrour, Rita 358