

*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*

Rocio Gonzalez-Diaz  
Maria-Jose Jimenez  
Belen Medrano (Eds.)

# Discrete Geometry for Computer Imagery

17th IAPR International Conference, DGCI 2013  
Seville, Spain, March 20-22, 2013  
Proceedings



Springer

Volume Editors

Rocio Gonzalez-Diaz  
Maria-Jose Jimenez  
Belen Medrano

University of Seville  
Applied Math I Department  
Avd. Reina Mercedes s/n  
41012 Seville, Spain

E-mail: {rogodi; majiro; belenmg}@us.es

ISSN 0302-9743 e-ISSN 1611-3349  
ISBN 978-3-642-37066-3 e-ISBN 978-3-642-37067-0  
DOI 10.1007/978-3-642-37067-0  
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013932474

CR Subject Classification (1998): I.3.5, I.4.1, I.4, I.3, I.5, G.2

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

© Springer-Verlag Berlin Heidelberg 2013

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 organization of the 17<sup>th</sup> International Conference on Discrete Geometry for Computer Imagery (DGCI 2013) has been a rewarding experience for our Andalusian research group, *Combinatorial Image Analysis (CIMAgroup)*. As organizers, we are pleased with the participation of many researchers from all around the world, taking into account the financial difficulties of our times. Indeed, submissions from 26 different countries confirm the international status of the conference.

This collection documents the contributions presented at DGCI 2013, which focus on geometric transforms, discrete and combinatorial tools for image segmentation and analysis, discrete and combinatorial topology, discrete shape representation, recognition and analysis, models for discrete geometry, morphological analysis, and discrete tomography.

Following a peer-reviewing process by at least two qualified reviewers, 34 papers were accepted, out of 56 submissions. Altogether, 22 papers were scheduled for oral presentation in single-track sessions, and 12 papers were presented as posters.

It has been a great honor for us to count on the participation of three internationally well-known researchers as invited speakers: Herbert Edelsbrunner (Professor at the Institute of Science and Technology, Vienna University, Austria), Francisco Escolano (Associate Professor at the University of Alicante, Spain), and Konrad Polthier (MATHEON-Professor and Director of the Mathematical Geometry Processing group at Freie Universität Berlin, Germany).

We would like to express our gratitude to the Reviewing and Program Committee members for their valuable comments, which enabled the authors to improve the quality of their contributions. We are also grateful to the Steering Committee for giving us the chance to organize this event and especially to David Coeurjolly for his support and helpful advice.

DGCI 2013 has been supported by the International Association of Pattern Recognition (IAPR). DGCI conferences are the main events associated with the Technical Committee on Discrete Geometry IAPR-TC18. This conference could not have been organized without our sponsoring institutions: University of Seville (Vice-rectorate for Research, Vice-rectorate of Internationalization, the Mathematics Institute IMUS, the Research and Teaching Foundation FIDETIA, Applied Math-I Department), Spanish Ministry of Economy and Competitiveness (Project MTM2012-32706), and European Science Foundation (ACAT program). We are also grateful to the School of Computer Engineering at the University of Seville, for hosting this event and providing all the necessary facilities.



Finally, our special thanks go to the local Organizing Committee for their valuable work and to all the participants attending the conference, who made this event a success.

March 2013

Rocio Gonzalez-Diaz  
Maria-Jose Jimenez  
Belen Medrano

# Organization

## Organizing Committee

Rocio Gonzalez-Diaz	University of Seville (Co-chair)
Maria-Jose Jimenez	University of Seville (Co-chair)
Javier Carnero	FIDETIA, University of Seville
Daniel Diaz	University of Seville
Desamparados Fernandez	University of Seville
Irene Fondon	University of Seville
Antonio Gutierrez	University of Seville
Belen Medrano	University of Seville
Helena Molina-Abril	University of Sheffield
Ana Maria Pacheco	University of Seville
Angel Tenorio	University of Pablo Olavide
Lidia de la Torre	University of Seville
Jose Antonio Vilches	University of Seville

## Steering Committee

Annick Montanvert (President)	University of Grenoble, France
Eric Andres	University of Poitiers, France
Gunilla Borgfors	University of Uppsala, Sweden
Srečko Brlek	University of Quebec, Canada
Jean-Marc Chassery	CNRS, University of Grenoble, France
David Coeurjolly	CNRS, University of Lyon, France
Isabelle Debled-Rennesson	LORIA, University of Nancy, France
Ullrich Köthe	University of Heidelberg, Germany
Kalman Palagyi	University of Szeged, Hungary
Gabriella Sanniti di Baja	Institute of Cybernetics Eduardo Caianiello, CNR, Italy

## Program Committee

Reneta Barneva	SUNY Fredonia (NY), USA
K. Joost Batenburg	Centrum Wiskunde & Informatica, Scientific Computing Group, The Netherlands
Isabelle Bloch	Telecom ParisTech-CNRS LTCI, France
Sara Brunetti	University of Siena, Italy

Michel Couprie	ESIEE/LIGM, France
Guillaume Damiand	LIRIS, CNRS, University of Lyon, France
Laurent Fuchs	SIC, University of Poitiers, France
Atsushi Imiya	University of Chiba, Japan
Christer Kiselman	Uppsala University, Sweden
Walter Kropatsch	TU Vienna, Austria
Jacques-Olivier Lachaud	University of Savoie, France
Rémy Malgouyres	LIMOS, University of Auvergne, Italy
Peer Stelldinger	University of Hamburg, Germany
Robin Strand	Centre for Image Analysis, Sweden
Edouard Thiel	LIF, Aix-Marseille University, France
Peter Veelaert	University College of Ghent, Belgium

## Reviewing Committee

Sylvie Alayrangues	Jean Philippe Domenger	Pascal Lienhardt
Andreas Alpers	Eric Domenjoud	Joakim Lindblad
Eric Andres	Ulrich Albert Eckhardt	Laurent Lucas
Dominique Attali	Philippe Even	Grégoire Malandain
Peter Attila Balazs	Thomas Fernique	Rémy Malgouyres
Antonio Bandera	Massimo Ferri	Jean-Luc Mari
Reneta Barneva	Fabien Feschet	Robert A. Melter
Nicolas Bedaride	Angel R. Frances Roman	Christian Mercat
Valerie Berthe	Patrizio Frosini	Serge Miguet
Gilles Bertrand	Laurent Fuchs	Annick Montanvert
Isabelle Bloch	Yan Gerard	Thierry Monteil
Olivier Bodini	Antonio Giraldo	Benoît Naegel
Gunilla Borgfors	Rocio Gonzalez-Diaz	Thanh Phuong Nguyen
Achille Braquelaira	Jeanpierre Guédon	Nicolas Normand
Valentin Brimkov	Lajos Hajdu	Kalman Palagyi
Srečko Brlek	Yll Haxhimusa	Nicolas Passat
Luc Brun	Atsushi Imiya	Samuel Peltier
Sara Brunetti	Maria Jose Jimenez	Christophe Picouleau
Sergio Cabello	Yukiko Kenmochi	Renzo Pinzani
Jean-Marc Chassery	Bertrand Kerautret	Xavier Provençal
David Coeurjolly	Christer Oscar Kiselman	Eric Remy
Michel Couprie	Reinhard Klette	Christian Ronse
Jean Cousty	Ullrich Koethe	Tristan Roussillon
Jose Crespo	T. Yung Kong	Gabriella Sanniti di Baja
Marie-Andrée Da Col	Walter G. Kropatsch	Isabelle Sivignon
Guillaume Damiand	Jacques-Olivier Lachaud	Peer Stelldinger
Isabelle	Claudia Landi	Robin Strand
Debled-Rennesson	Gaëlle Largeteau-Skapin	Mohamed Tajine
Pascal Desbarats	Bruno Eric Emmanuel	Hugues Talbot
Olivier Devillers	Levy	Alexandru Telea

Edouard Thiel  
Laure Tougne  
Jean-Luc Toutant

Sebastien Valette  
Peter Veelaert  
Laurent Vuillon

Laurent Wendling

## Sponsoring Institutions

DGCI 2013 was supported by the following institutions:

- Vice-rectorate of Research and Vice-rectorate of Internationalization, University of Seville
- European Science Foundation (ACAT program)
- Spanish Ministry of Economy and Competitiveness (Project MTM2012-326706)
- Mathematics Institute at the University of Seville (IMUS)
- International Association of Pattern Recognition (IAPR)
- Applied Math-I Department at University of Seville
- The Research and Teaching Foundation FIDETIA
- Spanish Network of Topology (RET)
- School of Computer Engineering at University of Seville

# The Complexity of Discrete Objects

Francisco Escolano

University of Alicante, Spain  
sco@dccia.ua.es

**Abstract.** In this paper we explore how to quantify the complexity of discrete objects (images, meshes, networks) which can be encoded in terms of graphs, digraphs or hypergraphs. Herein we present our Heat Flow-Thermodynamic Depth approach which combines ingredients of spectral graph theory, information theory and information projection. We illustrate the approach with several applications: image exploration (image complexity), mesh regularization and selection of optimal map functions in Extended Reeb Graphs (graph and digraph complexity) and structural categorization (hypergraph complexity).

## 1 Randomness vs. Statistical Complexity

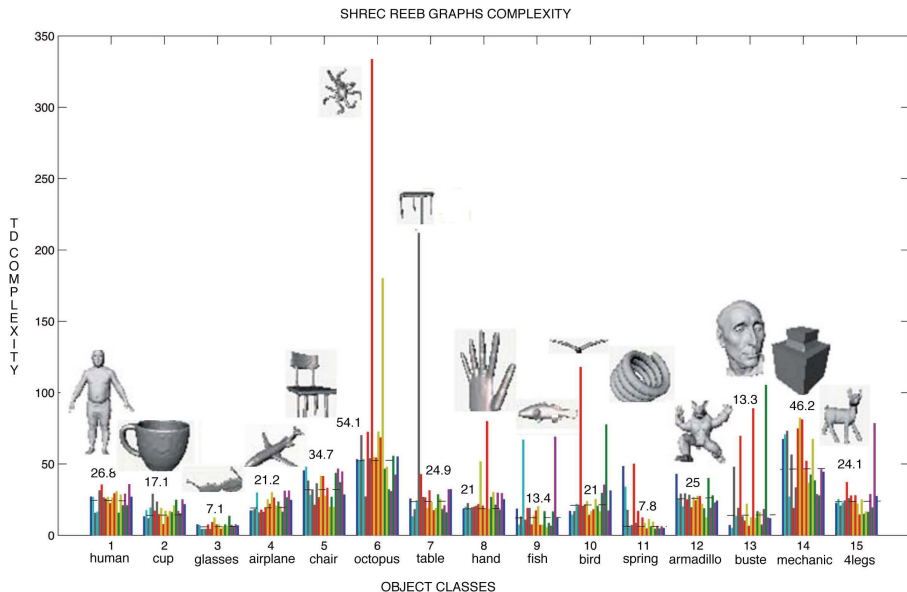
Given a discrete mathematical object (image, mesh, network) which can be encoded by a graph, digraph or hypergraph, the quantification of its *intrinsic complexity* plays a key role in understanding the underpinning structural principles shaping it. Such principles include: the information content of the encoding, the set of constraints over the information flowing through it, and the combinatorial exploration of the hypothesis that best explain its genesis. Information content may be posed in terms of estimating a given entropy. Information flow constraints may be discovered by probing the structure through heat kernels and wave equations. Generative hypothesis may rely on depth-based representations like logical depth or thermodynamic depth, hierarchical representations like grammars and compositional models, or dynamic rules like preferential attachment. The weight of each of the latter principles in the characterization of the encoding defines a *computable complexity measure* for it. For instance, strategies considering information content through Shannon or cross entropy are referred to as *randomness complexity* measures for they quantify the degree of disorganization (see a recent survey on graph entropy measures in [1]). On the other hand, methods relying on spectral measures, like the von Neumann entropy [2], or on Dirichlet sums, like Estrada's heterogeneity index are [3] more focused on quantifying the degree of regularity of the structure. These are good examples of the so called *statistical complexity* measures that vanished both for completely regular and completely random structures. Thermodynamic depth (TD) is a physics based approach [4] also belonging to the latter category. When dealing with graphs, thermodynamic depth aims to quantify how hard is to build a given graph (the macroscopic state) from scratch (microscopic states): if there

is a little uncertainty about the process and all the possible causal trajectories have a narrow variability, then the graph is narrow (simple); otherwise, when historical uncertainty arises and many causal trajectories have been excluded for reaching a given structural design, then the graph is deep (complex). Therefore, TD is purely based on the genesis principle. In this talk we present an statistical complexity measure which is focused on the connection between the second principle (information constraints) and the third one by *exploring the connection between heat diffusion and TD*. The underlying idea is that structure may impose constraints on heat diffusion. For instance, a complete graph must have zero complexity since there are no diffusion constraints. On the other hand, a linear graph imposes hard constraints. This occurs for both undirected and directed graphs. The combination of heat flow and TD breaks isospectrality. In [5] we present the main ingredients of the theory, embed both the von Neumann entropy and the Estrada's heterogeneity index in TD and show that the embedding of heat flow in TD outperforms the latter embeddings in terms of predicting the phylogeny of Protein-Protein Interaction (PPI) networks of several phyla of bacteria. In [6] we extended the so called *Heat Flow-TD complexity* to digraphs and compute the surface complexity of many European languages. In the following section we summarize some of the applications appealing to the DGCI community which are explored in the talk.

## 2 Applications of Structural Complexity

### 2.1 The Complexity of Images

We commence by motivating the need of measuring graph complexity beyond regularization or the minimization of description length. In [7] we propose an information-theoretic approach to understand eye-movement patterns, in relation to the task performed and image complexity. We analyze the distributions and amplitudes of eye-movement saccades, performed across two different image-viewing tasks: free viewing and visual search. Our working hypothesis is that the complexity of image information and task demands should interact. This should be reflected in the Markovian pattern of short and long saccades. We compute high-order Markovian models of performing a large saccade after many short ones and also propose a novel method for quantifying image complexity. The analysis of the interaction between high-order Markovianity, task and image complexity supports our hypothesis. Image complexity is measured in terms of computing the stationary distribution of a random walk in a grid-like structure whose nodes are image regions characterized by the responses of several filters. The edges in the grid are attributed by the mutual information between adjacent nodes. Given the stationary distribution we compute its Shannon entropy. Therefore, in this application we exploit a randomness complexity measure. To the best of our knowledge this is the first scientific connection between complexity quantification, perceptual tasks and image structure.



**Fig. 1.** SHREC complexities. TD complexity for each object in each class. For each class the dashed horizontal line and the number indicates the median TD complexity. Typical shapes (in classes with low complexity) and complex shapes (in classes with peaks).

## 2.2 The Complexity of Reeb Graphs

It is well known that the basic idea of Reeb graphs is to obtain information concerning the topology of a manifold  $M$  from information related to the critical points of a real function  $f$ . The Reeb graph produces a structure whose nodes are associated with the critical points of  $f$ . Here we follow the computational approach in [8], where a discrete counterpart of Reeb graphs, referred to as the *Extended Reeb Graph* (ERG), is defined for triangle meshes representing surfaces in  $\mathbf{R}^3$ . The basic idea underpinning ERG is to provide a region-based characterization of surfaces, rather than a point-oriented characterization. This is done by replacing the notion and role of critical points with that of *critical areas*, and the analysis of the behaviour of the level sets with the analysis of the behaviour of surface stripes, defined by partitioning the co-domain of  $f$  into a finite set of intervals. We consider in more detail a finite number of level sets of  $f$ , which divide the surface into a set of regions. Each region is classified as a *regular* or a *critical area* according to the number and the value of  $f$  along its boundary components. Critical areas are classified as *maximum*, *minimum* and *saddle* areas. A node in the graph is associated with each critical area. Then arcs between nodes are detected through an expansion process of the critical areas, by tracking the evolution of the level sets. A fundamental property of ERGs is their parametric nature with respect to the mapping function  $f$ : different choices of  $f$  produce different graphs. Also, the graphs inherit the invariance properties, if any, of the underlying mapping function. The mapping function

has to be selected according to the invariance and shape properties required for the task at hand. For instance, the analysis of the SHREC database from the point of view of barycenter Reeb graphs is summarized in Fig. 1. In this figure we represent the thermodynamic depth complexities of the 300 models and their variation among classes (median). Some classes are more heterogeneous than others. Therefore, thermodynamic depth of graphs seems to be a MDL (Minimum Description Length)-like measure of the mapping functions used for extracting Reeb graphs from 3D shapes. We will analyze several mapping functions and classify them in terms of intraclass variability (stability). This analysis concerns both undirected and directed graphs and both attributed and non-attributed ones. The spectral graph theory machinery developed for capturing the information flowing through Reeb graphs relies on the analysis of the combinatorial Laplacian and the evolution of heat kernels and quantum walks. Such elements are combined with the computation of node histories (TD) and also with Bregman divergences and information projection in order to quantify complexity.

### 2.3 The Complexity of Hypergraphs

Initial experiments with PPI networks show that complexity can be very helpful for structural classification purposes [9]. Recently, depth-based representations have been extended to characterize hypergraphs resulting in a high performance structural classifiers. Hypergraphs also allow us to mix several mapping functions for Reeb graphs which will be very helpful in their classification.

**Acknowledgements.** F. Escolano was funded by project TIN2012-32839 of the Spanish Government.

### References

- [1] Dehmer, M., Mowshowitz, A.: A History of Graph entropy Measures. *Information Sciences* 181, 57–78 (2011)
- [2] Passerini, F., Severini, S.: The von Neumann entropy of networks. arXiv:0812.2597v1, cond-mat.dis-nn (2008)
- [3] Estrada, E.: Quantifying network heterogeneity. *Phys. Rev. E* 82(6), 0066102 (2010)
- [4] Lloyd, S., Pagels, H.: Complexity as Thermodynamic Depth. *Ann. Phys.* 188, 186–213 (1988)
- [5] Escolano, F., Hancock, E.R., Lozano, M.A.: Heat Diffusion: Thermodynamic Depth Complexity of Networks. *Phys. Rev. E.* 85(3), 036206 (2012)
- [6] Escolano, F., Bonev, B., Hancock, E.R.: Heat Flow-Thermodynamic Depth Complexity in Directed Networks. In: *Proc. of SSPR/SPR*, pp. 190–198 (2012)
- [7] Bonev, B., Chuang, L.L., Escolano, F.: How do image complexity, task demands and looking biases influence human gaze behavior? *Pattern Recognition Letters* (to appear, 2013), doi:10.1016/j.patrec.2012.05.007
- [8] Biasotti, S.: Topological coding of surfaces with boundary using Reeb graphs. *Computer Graphics and Geometry* 7(1), 31–45 (2005)
- [9] Han, L., Escolano, F., Hancock, E.R., Wilson, R.C.: Graph characterizations from von Neumann entropy. *Pattern Recognition Letters* 33(15), 1958–1967 (2012)



# Stable Length Estimates of Tube-Like Shapes<sup>\*</sup>

Herbert Edelsbrunner<sup>1</sup> and Florian Pausinger<sup>2</sup>

IST Austria (Institute of Science and Technology Austria), Klosterneuburg, Austria,  
and Geomagic, Research Triangle Park, North Carolina

IST Austria (Institute of Science and Technology Austria), Klosterneuburg, Austria

**Abstract.** Mathematical objects can be measured unambiguously, but not so objects from our physical world. Even the total length of tube-like shapes has its difficulties. We introduce a combination of geometric, probabilistic, and topological methods to design a stable length estimate for tube-like shapes; that is: one that is insensitive to small shape changes.

## 1 Introduction

The length of a curve in Euclidean space is an elementary geometric concept, and it is well defined provided the curve is not wild. We consider the problem of computing the length of curve- or tube-like shapes, such as root systems of plants. Branching is allowed, but the real difficulty lies in the small but positive thickness, which renders length an undefined concept, at least in the mathematical sense. One may want to construct a 1-dimensional skeleton and take the length of that, but this construction is instable; see [1, 5]. Instead of stabilizing the skeleton, we aim at estimating the length of a hypothetical skeleton, which we leave unspecified. The difficulty in the related case of a coast line, studied famously by Mandelbrot [12], is the dependence on the resolution to which the curve is being measured. The length diverges as the resolution increases, suggesting the dimension of the coast line be larger than 1.

Noticing the abundance of tube-like shapes in nature and therefore in the sciences, we aim at producing a length estimate that is stable under perturbations of the shape. We believe that this will be useful in the study of geographic structures, including river and road networks, as well as biological and medical structures, including root systems of plants, blood vessels, nerve cells, and more. Our length estimation algorithm combines intuitive geometric ideas with topological methods:

---

<sup>\*</sup> This research is partially supported by the National Science Foundation (NSF) under grant DBI-0820624, the European Science Foundation (ESF) under the Research Network Programme, and the Russian Government under the Mega Project 11.G34.31.0053.

1. using the formula of Weyl [8, 14, 15], it expresses the length of a tube-like shape by an integral geometric representation of the second Quermassintegral;
2. applying a recent version of the Koksma-Hlawka Theorem [9–11], it approximates the resulting integral with explicit bounds on the integration error;
3. exploiting insights into the persistence diagram of a height function [2, 3, 7], it gives a length estimate that is stable under perturbations.

We implement the algorithm and analyze its performance. Our experiments give clear evidence for the effectivity of the topological method and the stability of the length estimate provided by our algorithm.

## 2 Tubes and Integral Geometry

We study and extend special cases of the tube formula of Weyl [8, 13, 15]. This formula holds for general smooth submanifolds of a finite-dimensional Euclidean space. The main result of this section is a simple relationship between the geometric properties of a curve and its thickened version in  $\mathbb{R}^3$ . Letting  $r_0$  be the thickening radius, we denote the thickened curve by  $\mathbb{M}$ . We have

$$L = Q_2/\pi; \tag{1}$$

$$L = [Q_2 - k(2\pi - 4)r_0 - 4\pi r_0]/\pi, \tag{2}$$

where  $L$  is the length of the curve, which is closed in (1), and has  $k \geq 0$  right-angle forks and  $k + 2$  tips in (2). For  $k = 0$ , we have a curve with two endpoints. To define  $Q_2$ , we introduce

$$Q_i(\mathbb{M}) = c_i \cdot \int_P \chi(\mathbb{M} \cap P) dP, \tag{3}$$

called the  $i$ -th *Quermassintegral* over the set of  $i$ -dimensional planes,  $P$ , in which  $c_i$  is a constant independent of  $\mathbb{M}$ . For  $i = 2$ , we have  $c_i = 1$ , and  $Q_2 = Q_2(\mathbb{M})$  is the total mean curvature of the bounding surface.

## 3 Quasi Randomness

In order to evaluate the second Quermassintegral, we apply a version of the Koksma-Hlawka Theorem recently proved by Harman [9]. This theorem bounds the integration error,

$$\text{Err}(N, X) = \left| \frac{1}{N} \sum_{j=1}^N F(x_j) - \int F(x) dx \right|, \tag{4}$$

in which  $X = \{x_1, x_2, \dots, x_N\}$  denotes a finite point set. It separates the contributions of the variation of the function and the distribution properties of the points at which the function is evaluated.

## 4 Persistence and Stability

We modify the straight length estimation formulas to get stable estimates for tube-like shapes. The tool for this purpose is persistent homology; see e.g. [6]. The main result is an expression of stability for the damped persistence moments of a function:

**1 (Damped Stability Theorem for Tubes)** *Let  $\mathbb{M}$  be a tube of radius  $r_0$  in  $\mathbb{R}^3$ , let  $f, g : \mathbb{M} \rightarrow \mathbb{R}$ , and set  $C = 4 + \delta$ . Then for every dimension  $0 \leq p \leq 2$ , every direction  $u \in \mathbb{S}^2$ , and every  $\delta > 0$ , we have*

$$|X_p^C(f) - X_p^C(g)| \leq \text{const} \cdot \frac{L^{1+\delta}}{r_0^{1+\delta}} \cdot \|f - g\|_\infty. \quad (5)$$

Here  $X_p^C(f)$  is related to the persistence moments, as we now explain. First, introduce

$$B_p^k(f) = \sum_{A \in \text{Up}_p(f)} \text{pers}(A)^k + \sum_{A \in \text{Dn}_{p+1}(f)} \text{pers}(A)^k, \quad (6)$$

in which the two sums are over the points in the persistence diagrams of the function. Specifically,  $\text{Up}_p(f)$  consists of all points in the  $p$ -dimensional diagram whose birth-coordinates are smaller than the death-coordinates, and  $\text{Dn}_{p+1}(f)$  consists of all points in the  $(p+1)$ -dimensional diagram whose birth-coordinates are larger than the death-coordinates. We call  $B_p^k(f)$  the  $k$ -th  $p$ -dimensional persistence moment. For  $k = 1$ , its significance lies in the relationship to the second Quermassintegral:

$$Q_2(\mathbb{M}) = \frac{1}{2} \int_{u \in \mathbb{S}^2} \left( \sum_{p=0}^2 (-1)^p B_p^1(f_u) \right) du, \quad (7)$$

where  $f_u$  is the height function on  $\mathbb{M}$  in the direction  $u$ . We split  $B_p^k(f) = B_p^k(f, r_0^-) + B_p^k(f, r_0^+)$  by collecting the values for  $\text{pers}(A) < r_0$  in the first term and values for  $\text{pers}(A) \geq r_0$  in the second term. For  $C = 4 + \delta$ , we set

$$X_p^C(f) = B_p^1(f, r_0^+) + \frac{1}{r_0^{C-1}} \cdot B_p^C(f, r_0^-), \quad (8)$$

$$\bar{Q}_2(\mathbb{M}) = \frac{1}{2} \int_{u \in \mathbb{S}^2} \left( \sum_{p=0}^2 (-1)^p X_p^C(f_u) \right) du, \quad (9)$$

calling  $\bar{Q}_2 = \bar{Q}_2(\mathbb{M})$  the *stabilized mean curvature* of  $\mathbb{M}$  at scale  $r_0$ . The Damped Stability Theorem now implies a similar expression of stability for the stabilized mean curvature.

## 5 Computational Experiments

We describe experimental results for the algorithms implementing the mathematical formulas developed in the preceding sections. We test accuracy as well as stability on small datasets, for which the answers are known, and investigate speed of convergence on root system data. We use three different algorithms to compute or approximate the total mean curvature of the boundary of a polytope  $\mathbb{M}$  in  $\mathbb{R}^3$ , and to estimate the length of  $\mathbb{M}$ :

- *Discrete Mean Curvature* (DMC): we compute the total mean curvature as half the sum over all boundary edges of the length times the angle between the two adjacent face normals; see e.g. [4].
- *Plane Sampling* (PS): we approximate the total mean curvature by summing up the Euler characteristics of the intersections between  $\mathbb{M}$  and planes sampled in  $\mathbb{R}^3$ .
- *Direction Sampling* (DS): we approximate the total mean curvature by summing up the alternating persistence moments of height functions defined by sampled directions on the 2-sphere.

The result of the DMC Algorithm is the total mean curvature of  $\mathbb{M}$  up to machine precision, which we use as the baseline for comparisons. The result of the PS Algorithm converges to the total mean curvature, and we get an impression of the speed of convergence from a comparison with the precise measurement. The basic version of the DS Algorithm is a reformulation of the PS Algorithm, but it offers the opportunity to filter out low-persistence contributions, thus stabilizing the length estimate; see the Damped Stability Theorem of the previous section. Indeed, the design of the algorithm implementing (9) is the main achievement of this paper. Experimental results comparing the performance of this algorithm with others can be found in the full version of this paper.

## References

1. 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)
2. Cohen-Steiner, D., Edelsbrunner, H.: Inequalities for the curvature of curves and surfaces. *Found. Comput. Math.* 7, 391–404 (2007)
3. Cohen-Steiner, D., Edelsbrunner, H., Harer, J.: Stability of persistence diagrams. *Discrete Comput. Geom.* 37, 103–120 (2007)
4. Cohen-Steiner, D., Morvan, J.-M.: Restricted Delaunay triangulations and normal cycles. In: *Proc. 19th Ann. Sympos. Comput. Geom.*, pp. 312–321 (2003)
5. De Floriani, L., Spagnuolo, M. (eds.): *Shape Analysis and Structuring*. Springer, Berlin (2008)
6. Edelsbrunner, H., Harer, J.L.: *Computational Topology. An Introduction*. Amer. Math. Soc., Providence, Rhode Island (2010)
7. Edelsbrunner, H., Letscher, D., Zomorodian, A.J.: Topological persistence and simplification. *Discrete Comput. Geom.* 28, 511–533 (2002)

8. Gray, A.: Tubes. Addison-Wesley, Redwood City (1990)
9. Harman, G.: Variations on the Koksma-Hlawka inequality. *Unif. Distrib. Theory* 5, 65–78 (2010)
10. Hlawka, E.: Funktionen von beschränkter Variation in der Theorie der Gleichverteilung. *Ann. Math. Pura Appl.* 54, 325–333 (1961)
11. Koksma, J.F.: Een algemeene stelling inuit de theorie der gelijkmatige verdeeling modulo 1. *Mathematica (Zutphen B)* 11, 7–11 (1942-1943)
12. Mandelbrot, B.: How long is the coast of Britain? Statistical self-similarity and fractional dimension. *Science* 156, 636–638 (1967)
13. Morvan, J.-M.: Generalized Curvatures. Springer, Berlin (2008)
14. Steiner, J.: Über parallele Flächen. *Monatsber. Akad. Wiss. Berlin*, 114–118 (1840); also *Werke* 2, 171–178 (1882)
15. Weyl, H.: On the volume of tubes. *Am. J. Math.* 61, 461–472 (1939)

# Table of Contents

## Models for Discrete Geometry

Optimal Covering of a Straight Line Applied to Discrete Convexity . . . . .	1
<i>Jean-Marc Chassery and Isabelle Sivignon</i>	
On Dimension Partitions in Discrete Metric Spaces . . . . .	11
<i>Fabien Rebatel and Édouard Thiel</i>	
Walking in the Farey Fan to Compute the Characteristics of a Discrete Straight Line Subsegment . . . . .	23
<i>Isabelle Sivignon</i>	
Persistent Patterns in Integer Discrete Circles . . . . .	35
<i>André Hoarau and Thierry Monteil</i>	
Comparison of Point Clouds Acquired by 3D Scanner . . . . .	47
<i>Natalia Dyshkant</i>	
Generalized Simple Surface Points . . . . .	59
<i>J.C. Ciria, E. Domínguez, A.R. Francés, and A. Quintero</i>	

## Discrete and Combinatorial Topology

A Parallel Thinning Algorithm for Grayscale Images . . . . .	71
<i>Michel Couprie, Nivando Bezerra, and Gilles Bertrand</i>	
New Structures Based on Completions . . . . .	83
<i>Gilles Bertrand</i>	
Asymptotic Analysis and Random Sampling of Digitally Convex Polyominoes . . . . .	95
<i>O. Bodini, Ph. Duchon, A. Jacquot, and L. Mutafchiev</i>	
Critical Connectedness of Thin Arithmetical Discrete Planes . . . . .	107
<i>Valérie Berthé, Damien Jamet, Timo Jolivet, and Xavier Provençal</i>	
A 3D Curvilinear Skeletonization Algorithm with Application to Path Tracing . . . . .	119
<i>John Chaussard, Laurent Noël, Venceslas Biri, and Michel Couprie</i>	

## Geometric Transforms

From the Zones of Influence of Skeleton Branch Points to Meaningful Object Parts . . . . .	131
<i>L. Serino, C. Arcelli, and G. Sanniti di Baja</i>	
Merging Faces: A New Orthogonal Simplification of Solid Models . . . . .	143
<i>Irving Cruz-Matías and Dolores Ayala</i>	
Sufficient Conditions for Topological Invariance of 2D Images under Rigid Transformations . . . . .	155
<i>Phuc Ngo, Yukiko Kenmochi, Nicolas Passat, and Hugues Talbot</i>	
Digital Distances and Integer Sequences . . . . .	169
<i>Nicolas Normand, Robin Strand, and Pierre Evenou</i>	

## Discrete Shape Representation, Recognition and Analysis

The Persistence Space in Multidimensional Persistent Homology . . . . .	180
<i>Andrea Cerri and Claudia Landi</i>	
A Study of Monodromy in the Computation of Multidimensional Persistence . . . . .	192
<i>Andrea Cerri, Marc Ethier, and Patrizio Frosini</i>	
Skeleton Extraction of Vertex Sets Lying on Arbitrary Triangulated 3D Meshes . . . . .	203
<i>Dimitri Kudelski, Sophie Viseur, and Jean-Luc Mari</i>	
Integral Based Curvature Estimators in Digital Geometry . . . . .	215
<i>David Coeurjolly, Jacques-Olivier Lachaud, and Jérémy Levallois</i>	
A Fast Algorithm for Liver Surgery Planning . . . . .	228
<i>Fajie Li, Xinbo Fu, Gisela Klette, and Reinhard Klette</i>	
$O(n^3 \log n)$ Time Complexity for the Optimal Consensus Set Computation for 4-Connected Digital Circles . . . . .	241
<i>Gaelle Largeteau-Skapin, Rita Zrour, and Eric Andres</i>	
Efficient Robust Digital Annulus Fitting with Bounded Error . . . . .	253
<i>Minh Son Phan, Yukiko Kenmochi, Akihiro Sugimoto, Hugues Talbot, Eric Andres, and Rita Zrour</i>	
Arc Recognition on Irregular Isothetic Grids and Its Application to Reconstruction of Noisy Digital Contours . . . . .	265
<i>Jean-Luc Toutant, Antoine Vacavant, and Bertrand Kerautret</i>	

## Morphological Analysis and Discrete Tomography

Reconstruction of Quantitative Properties from X-Rays . . . . .	277
<i>Fatma Abdmouleh and Mohamed Tajine</i>	
On the Non-additive Sets of Uniqueness in a Finite Grid . . . . .	288
<i>Sara Brunetti, Paolo Dulio, and Carla Peri</i>	
On the Degree Sequences of Uniform Hypergraphs . . . . .	300
<i>Andrea Frosini, Christophe Picouleau, and Simone Rinaldi</i>	
How to Decompose a Binary Matrix into three <i>hv</i> -convex Polyominoes . . . . .	311
<i>Andrea Frosini and Christophe Picouleau</i>	
Multi-resolution Cell Complexes Based on Homology-Preserving Euler Operators . . . . .	323
<i>Lidija Čomić, Leila De Floriani, and Federico Iuricich</i>	

## Discrete and Combinatorial Tools for Image Segmentation and Analysis

Convergence of Level-Wise Convolution Differential Estimators . . . . .	335
<i>Damien Gonzalez, Rémy Malgouyres, Henri-Alex Esbelin, and Chafik Samir</i>	
Concurrency Relations between Digital Planes . . . . .	347
<i>Peter Veelaert, Maarten Stembrouck, and Dirk Van Haerenborgh</i>	
Scale Filtered Euclidean Medial Axis . . . . .	360
<i>Michał Postolski, Michel Couprie, and Marcin Janaszewski</i>	
A Method for Feature Detection in Binary Tomography . . . . .	372
<i>Wagner Fortes and K. Joost Batenburg</i>	
Knot Segmentation in Noisy 3D Images of Wood . . . . .	383
<i>A. Krähenbühl, Bertrand Kerautret, and I. Debled-Rennesson</i>	
Multigrid Convergent Curvature Estimator . . . . .	395
<i>Christophe Fiorio, Christian Mercat, and Frédéric Rieux</i>	
<b>Author Index</b> . . . . .	407



# Optimal Covering of a Straight Line Applied to Discrete Convexity

Jean-Marc Chassery and Isabelle Sivignon

**gipsa-lab**, CNRS, UMR 5216, F-38420, France

{jean-marc.chassery,isabelle.sivignon}@gipsa-lab.grenoble-inp.fr

**Abstract.** The relation between a straight line and its digitization as a digital straight line is often expressed using a notion of proximity. In this contribution, we consider the covering of the straight line by a set of balls centered on the digital straight line pixels. We prove that the optimal radius of the balls is strictly less than one, and can be expressed as a function of the slope of the straight line. This property is used to define discrete convexity in concordance with previous works on convexity.

## 1 Introduction

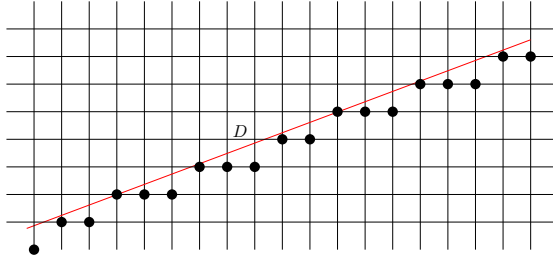
From the seminal work of Sklansky [12], discrete convexity has been the subject of many studies with a common objective: transcribe the Euclidean definition in the digital space.

In the eighties, a previous work introduced  $\varepsilon$ -convexity using covering of connected sets by balls [1]. In the case of convex shapes, it was shown in [1] that the value  $\varepsilon$  can be written as  $\frac{p}{p+1}$  (lower than 1) where  $p$  is a parameter computed from the edges of the convex hull of the shape. However, the value of this parameter  $p$  was given algorithmically but not analytically. In this paper, using the arithmetical definition of a digital straight line [9,3], we prove that  $\varepsilon$  can be expressed exactly as a function of the characteristics of the digital straight line supporting the edges of the convex hull of the shape.

After classical definitions of digital straight line and its characteristics, we establish the property of covering a straight line by balls centered exclusively on digital straight line pixels. The following section applies that property to the case of discrete convexity and we present the algorithm for discrete convex hull computation working only on digital space.

## 2 Preliminary Definitions

Let  $\mathcal{L}$  be an Euclidian straight line in  $\mathbb{R}^2$  given by the equation  $ax - by + \mu = 0$ , with  $a, b, \mu$  in  $\mathbb{Z}$ , and  $\gcd(a, b) = 1$ . In the following, we also assume without loss of generality that  $0 \leq a \leq b$ . All other cases are symmetrical. Such a straight line may be considered as supporting a linear contour of a shape, such that all the points of the shape are on the same side of the straight line.



**Fig. 1.** A straight line  $\mathcal{L}$  of equation  $3x - 8y + \mu = 0$  and its OBQ digitization analytically given by the digital straight line of equation  $0 \leq 3x - 8y + \mu < 8$

Let us now consider the Object Boundary Quantization of  $\mathcal{L}$  on the isothetic regular grid. It is given by the set of pixels  $(x, y)$  such that  $x \in \mathbb{Z}$  and  $y = \lfloor \frac{-ax - \mu}{b} \rfloor$  (see Figure 1).

It is well known that this set of digital points denoted by  $L$  is a simple 8-connected digital straight line (DSL) that can be defined by the diophantine equation [9,3]  $0 \leq ax - by + \mu < \max(|a|, |b|) = b$ .

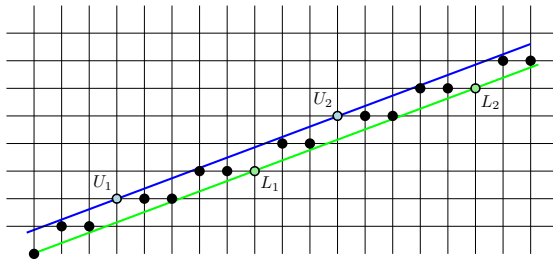
The slope of  $L$  is  $\frac{a}{b}$ ,  $\mu$  is the shift at origin. The remainder of  $L$  for a given digital point  $(x, y)$  is the value  $ax - by + \mu$ .

Two particular straight lines are associated to this DSL  $L$ :

- the upper leaning line given by equation  $ax - by + \mu = 0$  and,
- the lower leaning line given by equation  $ax - by + \mu = b - 1$ .

The digital points lying on these lines are similarly called leaning points (see Figure 2). Upper leaning points have remainder value 0 while lower leaning points have remainder value  $b - 1$  (see [2] for more details).

Since  $a, b, \mu$  are integers and  $\mathcal{L}$  is digitized with OBQ,  $\mathcal{L}$  is identified as the upper leaning line of  $L$ .



**Fig. 2.** Digital straight line of equation  $0 \leq 3x - 8y + \mu < 8$  with upper (lower) leaning points  $U_i$  ( $L_i$ )

### 3 Optimal Covering of a Straight Line

#### 3.1 Setting the Problem

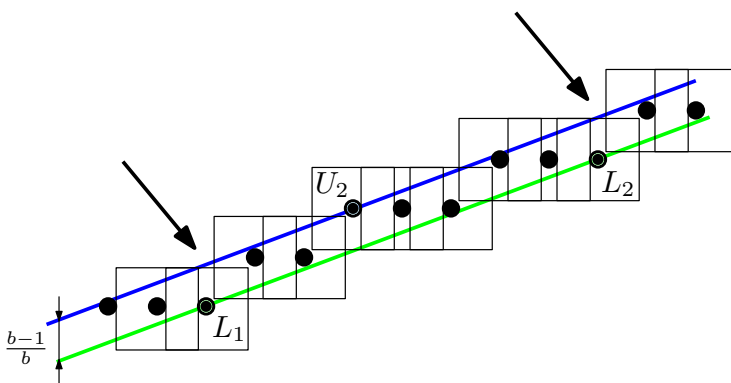
The objective is to cover the straight line  $\mathcal{L}$  with closed balls centered on the points of the DSL  $L$ . Moreover, the union of these balls shall not contain any other digital point. These balls are defined according to the  $L_\infty$  metric such that the ball of radius  $\varepsilon$  is defined by  $B(P, \varepsilon) = \{q | d_\infty(P, q) \leq \varepsilon\}$ . In our framework, we assume that the radius  $\varepsilon$  is the same for all the points of  $L$  (see Figure 3).

In the general case, setting  $\varepsilon$  to  $\frac{1}{2}$  is not sufficient to cover the straight line  $\mathcal{L}$  except for the special cases of horizontal, vertical and diagonal lines. We can notice the following elementary property:

*Property 1.* The band delimited by the 2 leaning lines (upper and lower) has a vertical thickness of  $\frac{b-1}{b}$ .

The proof is straightforward from the equations of the leaning lines. Figure 3 illustrates the covering of a straight line of slope  $\frac{a}{b}$  by balls of radius  $\frac{b-1}{b}$ . We remark that this radius is not sufficient since parts of the straight line remain uncovered.

To analyze this covering, we proceed by successive couples of adjacent pixels. It is easy to verify that the vertical distance between a pixel of remainder  $r$  of the DSL and the line  $\mathcal{L}$  is equal to  $r/b$ , with  $r$  varying from 0 to  $b-1$ . The value  $\frac{b-1}{b}$  is obtained for a lower leaning point. Let us consider two successive pixels of  $L$  such that the first one is a lower leaning point. Then we notice that the union of the balls of radius  $\frac{b-1}{b}$  centered on these points leaves a part of  $\mathcal{L}$  uncovered (see Figure 3). So the minimum  $\varepsilon$  for the line  $\mathcal{L}$  to be covered is greater than  $\frac{b-1}{b}$ .



**Fig. 3.** Balls of radius  $\varepsilon = \frac{b-1}{b}$  centered on the DSL points do not cover  $\mathcal{L}$ : uncovered regions are indicated by arrows

In the following, we denote by  $P_0, P_1, \dots, P_n$  the ordered sequence (increasing abscissae) of pixels of the DSL  $L$ . For each pixel  $P_i$  we can define as  $proj(P_i)$  the vertical projection of  $P_i$  on the straight line  $\mathcal{L}$ . Consequently, the Euclidean straight line  $\mathcal{L}$  may be partitionned as the union of subsegments  $[proj(P_i), proj(P_{i+1})]$ .

*Property 2.* If for every pair of successive pixels  $P_i$  and  $P_{i+1}$ ,  $B(P_i, \varepsilon) \cup B(P_{i+1}, \varepsilon)$  covers the straight segment  $[proj(P_i), proj(P_{i+1})]$  then the straight line  $\mathcal{L}$  is covered by the union of the balls  $B(P_i, \varepsilon)$ .

As said before,  $\varepsilon$  must be greater than  $\frac{b-1}{b}$ , and it is easy to see that it must also be strictly lower than 1, otherwise new digital points are included in the union of balls.

### 3.2 Optimal Covering

The following theorem defines the optimal value of  $\varepsilon$  as a function of the straight line parameters  $a$  and  $b$ .

**Theorem 1.** *Let  $\mathcal{L}$  a straight line of equation  $ax - by + \mu = 0$  and  $L$  its digitization with the OBQ scheme. In the context of the previous definitions, the union of balls  $B(P_i, \varepsilon)$  centered on pixels of the DSL  $L$  with radius  $\varepsilon = \max(\frac{1}{2}, \frac{|a|+|b|-1}{|a|+|b|})$  covers the straight line  $\mathcal{L}$ . This set doesn't contain any other digital pixels excepted those of the DSL.*

*Proof.* We suppose that  $0 \leq a \leq b$ . First of all, if the parameters of  $\mathcal{L}$  are  $(0, 1, \mu)$  (horizontal straight line) or  $(1, 1, \mu)$  (diagonal straight line), the optimal value of  $\varepsilon$  is trivially equal to  $\frac{1}{2}$ . Otherwise,  $b$  is greater than or equal to 2.

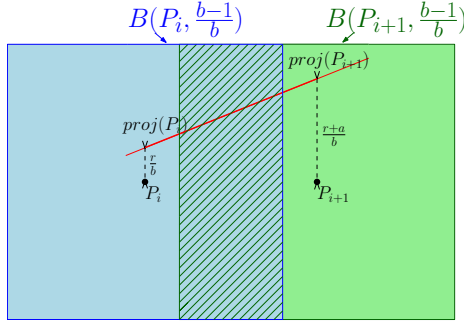
The distance between a point  $P_i$  and its projection  $proj(P_i)$  (see above) is equal to  $\frac{r}{b}$  if the remainder of  $P_i$  is equal to  $r$ .

In the case “ $P_i$  and  $P_{i+1}$  4-connected”, if  $r$  is the remainder of  $P_i$  then  $r + a$  is the remainder of  $P_{i+1}$ . Consequently, the distance between  $P_{i+1}$  and its projection  $proj(P_{i+1})$  is equal to  $\frac{r+a}{b}$ . We have the following inequalities:

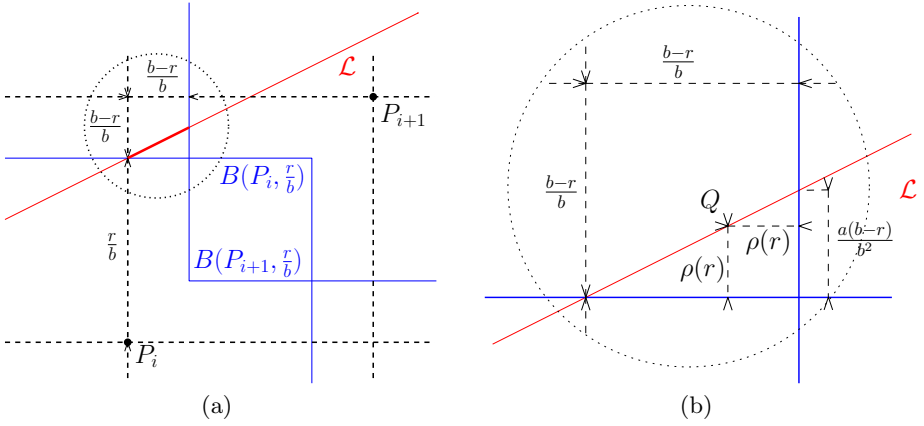
- $\frac{b-1}{b} > \frac{r}{b}$  ( $P_i$  belongs to  $L$ ),
- $\frac{b-1}{b} \geq \frac{r+a}{b}$  ( $P_{i+1}$  belongs to  $L$ ),
- $2\frac{b-1}{b} \geq 1$  (since  $b \geq 2$ ).

With the first two inequalities, we deduce that  $proj(P_i)$  and  $proj(P_{i+1})$  are covered by  $B(P_i, \frac{b-1}{b}) \cup B(P_{i+1}, \frac{b-1}{b})$ . The third inequality ensures that  $B(P_i, \frac{b-1}{b})$  and  $B(P_{i+1}, \frac{b-1}{b})$  overlap. We can conclude that the union of the two balls  $B(P_i, \frac{b-1}{b}) \cup B(P_{i+1}, \frac{b-1}{b})$  covers the segment  $[proj(P_i), proj(P_{i+1})]$  (see Figure 4).

In the case “ $P_i$  and  $P_{i+1}$  8-connected”, according to the Figure 5(b), we introduce the point  $Q$  belonging to the straight line  $\mathcal{L}$  situated at equal distance ( $L_\infty$  norm) from  $P_i$  and  $P_{i+1}$ . The point  $Q$  is on  $\mathcal{L}$  and shall belong to the two balls  $B(P_i, \varepsilon)$  and  $B(P_{i+1}, \varepsilon)$ . If  $r$  is the remainder of  $P_i$ , we denote by  $\varepsilon(r)$



**Fig. 4.** When  $P_i$  and  $P_{i+1}$  are 4-connected,  $B(P_i, \frac{b-1}{b}) \cup B(P_{i+1}, \frac{b-1}{b})$  covers the segment  $[proj(P_i), proj(P_{i+1})]$



**Fig. 5.** Illustration of the proof when  $P_i$  and  $P_{i+1}$  are 8-connected.  $r$  is the remainder of  $P_i$ . (b) is a close up of the region circled in (a).

the minimum radius such that  $B(P_i, \varepsilon(r)) \cup B(P_{i+1}, \varepsilon(r))$  covers the segment  $[proj(P_i), proj(P_{i+1})]$ . We can write  $\varepsilon(r)$  as the sum  $\frac{r}{b} + \rho(r)$ , with  $\rho(r) \geq 0$  (see Figure 5). Using Thalès theorem, we have

$$\frac{\rho(r)}{\frac{a(b-r)}{b^2}} = \frac{\frac{b-r}{b} - \rho(r)}{\frac{b-r}{b}}$$

We obtain

$$\rho(r) = \frac{a(b-r)}{b(a+b)}$$

Then  $\varepsilon(r) = \rho(r) + \frac{r}{b} = \frac{a+r}{a+b}$ .

$\varepsilon(r)$  is increasing and is maximum for  $r = b - 1$ , corresponding to  $\varepsilon = \frac{a+b-1}{a+b}$ .

This proof was developed in the case  $0 \leq a \leq b$ . It can be easily extended in the general case to obtain  $\varepsilon = \max(\frac{1}{2}, \frac{|a|+|b|-1}{|a|+|b|})$ . Since  $\varepsilon < 1$ ,  $P_i$  is the only digital point in the ball  $B(P_i, \varepsilon)$  for all  $i$ , which ends the proof.

## 4 Discrete Convexity

### 4.1 Definitions

An important literature has been developed for digital convexity [12,7,5,6,1]. A large number of these definitions are considered to be equivalent in case of simply connected sets [10,4]. The following definition is issued from the transcription of convexity definition in Euclidean space to discrete space. In Euclidean geometry, a region  $\mathcal{R}$  is convex if and only if for every pair of points  $p, q$  belonging to  $\mathcal{R}$ , the straight line segment  $[p, q]$  is included in  $\mathcal{R}$ . The following definition of discrete convexity replaces inclusion by covering with balls [1].

**Definition 1 ( $\varepsilon$ -convexity).** *A connected component  $S$  is  $\varepsilon$ -convex, with  $\varepsilon$  belonging to interval  $[\frac{1}{2}, 1[$  if:*

- for every pair of pixels  $P$  and  $Q$  of  $S$ ,
- for every real value  $\alpha$  belonging to  $]0, 1[$ ,

*there exists a pixel  $R$  belonging to  $S$  such that the point  $(\alpha P + (1 - \alpha)Q)$  of the straight line supported by the two points  $PQ$  belongs to the balls  $B(R, \varepsilon)$ , centered on  $R$  with radius  $\varepsilon$ .*

**Definition 2 (Discrete convexity).** *A connected component  $S$  is discrete convex if there exists a real  $\varepsilon \in [\frac{1}{2}, 1[$  such that  $S$  is  $\varepsilon$ -convex.*

### 4.2 Algorithmic Approach

The computation of the convex hull  $Conv(S)$  is done in two steps :

1. first, the  $x - y$  convex shape issued from  $S$  is computed;
2. then the convex hull is computed from the  $x - y$  convex shape.

The  $x - y$  convex shape of a connected component is defined as the convex shape along horizontal and vertical directions: for any points  $P$  and  $Q$  of the  $x - y$  convex shape, if  $P$  and  $Q$  are on the same line or column of the grid, then all the points between  $P$  and  $Q$  (on this line or column) also belong to the  $x - y$  convex shape. It is evident that the  $x - y$  convex shape is included in the convex hull.

Algorithm 1 describes how to compute the  $x - y$  convex shape. Suppose that the shape  $S$  is included in a binary image of size  $m \times n$ . The initialization step consists in:

1. compute the indices  $i_f$  and  $i_l$  of the upper and lower lines containing points of  $S$  respectively;
2. for each line  $i$  between  $i_f$  and  $i_l$ , compute  $min(i)$  and  $max(i)$  as the minimum and maximum indices of points of  $S$  on line  $i$ .

This initialization step is done in  $\mathcal{O}(nm)$ . Afterwards, the three loops of Algorithm 1 are performed to compute the maximal and minimal abscissas for each line of the  $x - y$  convex shape of  $S$ .

---

**Algorithm 1.**  $x - y$  convex shape computation of a set  $S$  of discrete points

---

```

1 for  $i$  from  $i_f$  to  $i_l$  do
     $min_{\downarrow}(i) = \min(min(i), min_{\downarrow}(i - 1))$  ; //  $min_{\downarrow}(i)$  is initially set to  $m$ 
    ;
     $max_{\downarrow}(i) = \max(max(i), max_{\downarrow}(i - 1))$  ; //  $max_{\downarrow}(i)$  is initially set to 0
    ;
2 for  $i$  from  $i_l$  to  $i_f$  do
     $min_{\uparrow}(i) = \min(min(i), min_{\uparrow}(i + 1))$  ; //  $min_{\uparrow}(i)$  is initially set to  $m$ 
    ;
     $max_{\uparrow}(i) = \max(max(i), max_{\uparrow}(i + 1))$  ; //  $max_{\uparrow}(i)$  is initially set to 0
    ;
3 for  $i$  from  $i_f$  to  $i_l$  do
     $min(i) = \max(min_{\downarrow}(i), min_{\uparrow}(i))$ ;
     $max(i) = \min(max_{\downarrow}(i), max_{\uparrow}(i))$ ;
     $j_f = \min(j_f, min(i))$  ; //  $j_f$  is initially set to  $n$ 
    ;
     $j_l = \max(j_l, max(i))$  ; //  $j_l$  is initially set to 0
    ;

```

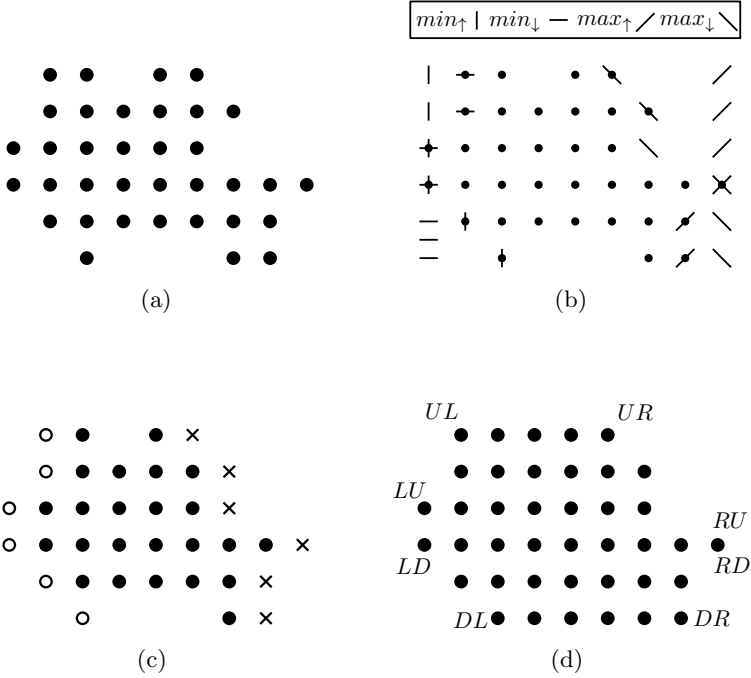
---

Figure 6(b) illustrates the first two loops of the algorithm (lines 1 and 2). Figure 6(c) illustrates the third loop (line 3).

At the end of Algorithm 1, the variables  $min(i)$  and  $max(i)$  contain the indices of the minimum and maximum points of the  $x - y$  convex shape for the line  $i$ . This algorithm also computes the coordinates  $i_f, i_l, j_f, j_l$  of the bounding rectangle of  $S$ .

Finally, the convex hull of  $S$  is computed as follows. To do so, we define the extremal point of the bounding rectangle as the points of  $S$  belonging to this rectangle and such that at least one of its two neighbours on the rectangle does not belong to  $S$ . These eight points are depicted on Figure 6(d).

Finally, the convex hull is computed from these extremal points using a technique similar to the algorithm of Sklansky [13][8](Chap. 13) on the polygon defined by the variables  $min(\cdot)$  and  $max(\cdot)$ . Note that this polygon is simple and completely visible from the outside, such that the algorithm works in this case. This algorithm works in  $\mathcal{O}(\max(m, n))$  time, which leads to a global complexity of  $\mathcal{O}(nm)$  to compute the convex hull of  $S$ . The result on the example of Figure 6 is given in Figure 7.



**Fig. 6.** (a) A discrete shape  $S$ . (b) Illustration of the variables  $min_{\downarrow}$ ,  $min_{\uparrow}$ ,  $max_{\downarrow}$ ,  $max_{\uparrow}$  used in Algorithm 1. (c) Illustration of the variables  $min(\cdot)$  and  $max(\cdot)$  at the end of Algorithm 1. (d) Extremal points of  $S$ .

### 4.3 Optimal Converging and Discrete Convexity

Let us denote by  $\{P_i, i = 0..n\}$  the ordered set of vertices of the convex hull  $Conv(S)$ . To each edge  $[P_i, P_{i+1}]$  we associate the slope parameters  $(a_i, b_i)$  such that  $\gcd(a_i, b_i) = 1$ . From Property 1, this edge is covered by balls centered on the pixels of its OBQ digitization with radius  $\max(\frac{1}{2}, \frac{|a_i|+|b_i|-1}{|a_i|+|b_i|})$ .

It is easy to prove that the connected set  $Conv(S)$  of discrete points included in  $Conv(S)$  is  $\varepsilon$ -convex with  $\varepsilon = \max_{i \in 0..n} \max(\frac{1}{2}, \frac{|a_i|+|b_i|-1}{|a_i|+|b_i|})$ .  $S$  is discrete convex if and only if it is equal to  $Conv(S)$ .

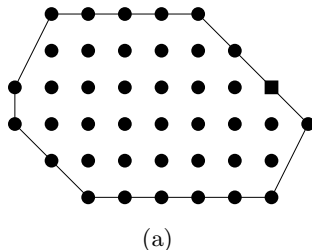
Moreover such definition is fully compatible with the continuous one as it is proved by the theorem [1]:

**Theorem 2.** *Let  $\mathcal{S}$  be a connected component in space  $\mathbb{R}^2$ .*

*If, for every sampling step, the discrete connected component  $S$  attached to  $\mathcal{S}$  is discrete convex, then  $\mathcal{S}$  is convex in space  $\mathbb{R}^2$ .*

Using  $S$  and  $Conv(S)$  a lot of features can be extracted to estimate a measure of convexity of  $S$  (number of missing pixels, distribution of concavities)[11]. These features are used to measure the degree of convexity of a shape in presence of many concavities.





**Fig. 7.** Convex hull of the discrete shape  $S$ : the point marked by a square is added during the application of Sklansky's algorithm. The discrete points belonging to the polygonal line are the vertices of  $Conv(S)$ .  $Conv(S)$  is defined as the discrete points inside  $Conv(S)$ .

## 5 Conclusion

The transcription of continuous concept like convexity into digital space needs specific attention for its definition as well as for its properties. In case of convexity we use the definition of digital straight line and we replace the notion of inclusion by the notion of covering. The main result obtained in this paper is issued from parametric representation of digital straight line to characterize the optimal radius for covering balls centered on digital straight line pixels.

## References

1. Chassery, J.M.: Discrete convexity: Definition, parametrization, and compatibility with continuous convexity. *Computer Vision, Graphics, and Image Processing* 21(3), 326–344 (1983)
2. Coeurjolly, D., Montanvert, A., Chassery, J.M.: *Géométrie discrète et images numériques. Hermès, traité IC2, série signal et image* (2007)
3. Debled-Rensson, I., Reveillès, J.P.: A linear algorithm for segmentation of digital curves. *International Journal of Pattern Recognition and Artificial Intelligence* 9(6), 635–662 (1995)
4. Eckhardt, U.: Digital Lines and Digital Convexity. In: Bertrand, G., Imiya, A., Klette, R. (eds.) *Digital and Image Geometry*. LNCS, vol. 2243, pp. 209–228. Springer, Heidelberg (2002)
5. Kim, C.E.: Digital convexity, straightness, and convex polygons 4(6), 618–626 (1982)
6. Kim, C.E., Rosenfeld, A.: Digital straight lines and convexity of digital regions 4(2), 149–153 (1982)
7. Kim, C.E., Sklansky, J.: Digital and cellular convexity. *Pattern Recognition* 15(5), 359–367 (1982)
8. Klette, R., Rosenfeld, A.: *Digital Geometry: Geometric methods for digital picture analysis*. Morgan Kaufmann (2004)
9. Reveillès, J.P.: *Géométrie discrète, calcul en nombres entiers et algorithmique*. Thèse d'état, Université Louis Pasteur, Strasbourg, France (1991)

10. Ronse, C.: A bibliography on digital and computational convexity (1961-1988) 11(2), 181–190 (1989)
11. Roussillon, T.: Algorithmes d'extraction de modèles géométriques discrets pour la représentation robuste des formes. Ph.D. thesis, Université Lumière Lyon 2 (2009)
12. Sklansky, J.: Recognition of convex blobs. *Pattern Recognition* 2(1), 3–10 (1970)
13. Sklansky, J.: Measuring concavity on a rectangular mosaic. *IEEE Transactions on Computers* (12), 1355–1364 (1972)

# On Dimension Partitions in Discrete Metric Spaces

Fabien Rebatel and Édouard Thiel

Laboratoire d'Informatique Fondamentale de Marseille (LIF, UMR 7279),  
Aix-Marseille Université,  
163 Avenue de Luminy, Case 901, 13288 Marseille cedex 9, France  
{Fabien.Rebatel,Edouard.Thiel}@univ-amu.fr

**Abstract.** Let  $(W, d)$  be a metric space and  $S = \{s_1 \dots s_k\}$  an ordered list of subsets of  $W$ . The distance between  $p \in W$  and  $s_i \in S$  is  $d(p, s_i) = \min\{d(p, q) : q \in s_i\}$ .  $S$  is a resolving set for  $W$  if  $d(x, s_i) = d(y, s_i)$  for all  $s_i$  implies  $x = y$ . A metric basis is a resolving set of minimal cardinality, named the metric dimension of  $(W, d)$ . The metric dimension has been extensively studied in the literature when  $W$  is a graph and  $S$  is a subset of points (classical case) or when  $S$  is a partition of  $W$ ; the latter is known as the partition dimension problem. We have recently studied the case where  $W$  is the discrete space  $\mathbb{Z}^n$  for a subset of points; in this paper, we tackle the partition dimension problem for classical Minkowski distances as well as polyhedral gauges and chamfer norms in  $\mathbb{Z}^n$ .

**Keywords:** dimension partition, metric dimension, distance geometry, discrete distance, norm.

## 1 Introduction

The partition dimension of a set is a combinatorial problem which generalizes the metric dimension of a set. These notions are classically studied in graph theory, and more generally in the field of distance geometry. Distance geometry is the characterization and study of sets of points based on the distance values between member pairs. Most of the notions of this field can be studied in the discrete space  $\mathbb{Z}^n$ , which adds digital and/or Euclidean geometry properties, as well as properties depending on the chosen distance. In this paper, we propose to tackle the partition dimension problem in  $\mathbb{Z}^n$  for some classical norms.

Let  $W$  be a (finite or infinite) set endowed with a metric  $d$ . We first consider the metric dimension and recall some results. Let  $S = (v_1, v_2, \dots, v_k)$  an ordered subset of points in  $W$ . The *representation* of  $p \in W$  with respect to  $S$  is the  $k$ -tuple  $r(p|S) = \{d(p, v_1), d(p, v_2), \dots, d(p, v_k)\}$ , also called the coordinates of  $p$ . The set  $S$  is a *resolving set* for  $W$  if every two elements of  $W$  have distinct representation. A resolving set having minimal cardinality is called a *metric basis* for  $W$ ; its cardinality is the *metric dimension*  $\dim(W)$  of  $W$ .

Harary and Melter gave in [7] the intrinsic metric dimension of any path, complete graph, cycle, wheel and complete bipartite graph. They also 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 [10] that all graphs having dimension 1 are paths and that there are non-planar graphs having dimension 2. The intrinsic metric dimension of trees can be efficiently solved in linear time, but finding the metric dimension of an arbitrary graph is NP-hard. The dimension of an arbitrary graph with  $n$  nodes can be approximated within a factor  $O(\log n)$  in polynomial time. Collections of bounds or exact values of metric dimensions are presented in [4][9]. For other results on graphs, see [1][2].

A number of results in digital geometry have also been established. Melter and Tomescu showed in [11] that when  $W$  is the digital plane, the metric bases are sets of three non-collinear points for the Euclidean distance  $d_2$ , whereas there are no finite metric bases for the city block distance  $d_1$  and the chessboard distance  $d_\infty$ . The  $d_1$  metric dimension of a  $n$ -dimensional axes-parallel rectangle is  $n$  [10], and the  $d_\infty$  metric dimension of a square is 3. If non axes-parallel rectangles are considered, there exists for both distances a rectangle in the digital plane such that its dimension is  $n$ , for any given  $n \geq 3$ . In [12], we have shown that the metric dimension in  $\mathbb{R}^n$  is infinite for any polyhedral (or partially polyhedral) central symmetric gauge; the metric dimension in  $\mathbb{Z}^n$  is also infinite for any chamfer norms. However, the metric dimension is finite in axes-parallel rectangles for both kind of spaces and distances.

The notion of partition dimension has been proposed in [5][6]. We recall the definition in a more general manner: let  $S = \{s_1 \dots s_k\}$  be an ordered list of subsets of  $W$ . The distance between  $p \in W$  and  $s_i \in S$  is  $d(p, s_i) = \min\{d(p, q) : q \in s_i\}$ . The representation of  $p$  with respect to  $S$  is  $r(p|S) = \{d(p, s_1), \dots, d(p, s_k)\}$ . As previously,  $S$  is a resolving set for  $W$  if every two elements of  $W$  have distinct representation.

When  $S$  is a set of points, we turn back again to the metric dimension problem. If  $S$  is a partition of  $W$ , we obtain new objects: a partition (metric) base is a resolving set of minimal cardinality, called the partition dimension  $\text{pd}(W)$ . For any nontrivial connected graph  $G$  we have  $\text{pd}(G) \leq \dim(G) + 1$  [5]. It should be noted that the partition dimension may be much smaller than the metric dimension [13]. In particular, the partition dimension of the digital plane with respect to  $d_1$  and  $d_\infty$  is 3 and 4, respectively. For other relationships between metric and partition dimension and graph diameter, as well as results on coloring and chromatic number, see [3].

The paper is organized as follows. In section 2, we define partition classes and study the combinatorics of the partition dimension problem. Then, in section 3, we present algorithms allowing to decide if a partition is resolving, and we evaluate their complexities. Next in section 4, we present specific properties of partition dimensions for gauges in Euclidean and digital spaces. We finally conclude in section 5.

## 2 Preliminaries

In this part, we reformulate the problem of partition dimension to understand the relationship between the latter and the metric basis problem. Moreover, we express their differences in combinatorics.

We first introduce a simple example, in which  $W$  is the set of elements  $(e_1, e_2, e_3, e_4, e_5)$ . Each partition can be expressed in several manners. For instance, the partition  $\{\{e_2\}, \{e_3, e_1, e_5\}, \{e_4\}\}$ , is equivalent to the partition  $\{\{e_1, e_3, e_5\}, \{e_2\}, \{e_4\}\}$ . To avoid ambiguity, we write each of them using the canonical notation defined as follows: in each part, the elements are sorted by their index; the parts are sorted by decreasing cardinal; parts having same cardinal are sorted in lexicographic order.

### 2.1 Partition Class

Given a partition we define its *partition class* as the vector of cardinalities of its parts. In our example, the partition  $\{\{e_1, e_3, e_5\}, \{e_2\}, \{e_4\}\}$  belongs to the class  $[3, 1, 1]$ .

The set of partition classes corresponds to the *integer partition*. The number of partitions of an integer  $n$  is given by the function  $p(n)$  [8]. The function  $p(n)$  grows extremely fast: as the first values for  $n = 1 \dots 9$  are 1, 2, 3, 5, 7, 11, 15, 22, 30, we have  $p(100) = 190\,569\,292$ .

Given a metric basis  $M = \{m_1, \dots, m_n\}$ , we can obtain a partition basis  $P = \{W \setminus M, \{m_1\}, \dots, \{m_n\}\}$  written in canonical form. The partition classes for metric basis problem are those which have at most one class with more than one element (i.e.  $[x, 1, \dots, 1]$ ,  $x \geq 1$ ). It only exists one class of partition representing a metric basis for each dimension of partition. This means that for a set of 100 points, there are only 100 classes of metric basis among the 190 millions available classes.

**Definition 1.** *The distance between two classes of partition is the sum of the difference between the coordinates of their vectors:*

$$d(C_1, C_2) = \sum_i |C_1[i] - C_2[i]|.$$

If both vectors do not have the same number of part, the smallest one is completed by parts of size zero. The coordinate vectors are completed by zeros if needed. The distance between classes of partitions is a  $\ell_1$  distance between vector coordinates. We remark that two partition classes with a distance of 1 do not exist for a given number of points. Indeed, the only way to have a distance of 1 between two classes is to have only one part where the number of points is different, so the total number of points is not the same.

Two classes are called *adjacent* if their distance is 2, which means that there are only two parts whose cardinals differ, one gaining a point and one missing a point.

We define a distance between two partitions as follows:

**Definition 2.** *The distance between two partitions is given by the Hamming distance of their coordinate vectors.*

Once again, if partitions do not have the same size, the smallest one is padded by empty parts (Hamming distance is defined for two vectors of the same size). Two partitions will be named *adjacent* if one of the properties is fulfilled:

- i) their partition distance is 1;
- ii) the partition class distance is 0 and the partition distance is 2.

## 2.2 Counting

The number of partitions of  $n$  labeled objects into  $k$  non-empty unlabeled subsets is given by the *Stirling number of the second kind*  $\left\{ \begin{smallmatrix} n \\ k \end{smallmatrix} \right\}$  (see sequence A008277 in OEIS):

$$\left\{ \begin{smallmatrix} n \\ k \end{smallmatrix} \right\} = \frac{1}{k!} \sum_{j=0}^k (-1)^{k-j} \binom{k}{j} j^n .$$

This number is important to evaluate the difficulty to find a resolving partition of  $n$  points for a given dimension  $k$ . If we look only at metric basis classes, the number of partitions is reduced to the binomial coefficient  $\binom{n}{k}$ .

Now in the case where we need to enumerate all of the possible partitions (up to size  $n$ ), the cardinal will be the sum

$$B_n = \sum_{k=0}^n \left\{ \begin{smallmatrix} n \\ k \end{smallmatrix} \right\} ,$$

known as the *Bell number*  $B_n$  (see sequence A000110 in OEIS).

The first 10 numbers of Bell are 1, 1, 2, 5, 15, 52, 203, 877, 4140, 21147; we have  $B(20) = 5\,832\,742\,205\,057$ , and  $B(100)$  is given by a 116 digital number. The number of possible metric basis is given by the sum of the binomial coefficient  $\sum_{i=0}^n \binom{n}{i} = 2^n$ . As a comparison,  $2^{20} = 1\,048\,576$  is much smaller than  $B(20)$ .

An other interesting comparison can be made between classes of the same dimension. The number of partition in each classes can be computed by this recursive function:

$$\#[p_i, p_{i+1}, \dots, p_k] = \frac{1}{A_i} \left( \sum_{j=i}^k p_j \right) \#[p_{i+1}, \dots, p_k] ,$$

where  $A_i$  is the number of parts in  $[p_i, p_{i+1}, \dots, p_k]$  which value equals  $p_i$ .

The number of classes of dimension  $n$  is highly unevenly distributed; the cardinal of the metric bases class is minimal and is far smallest than the cardinal of the other classes.

## 3 Algorithm

In this section we describe some methods to determine if a given partition is resolving. We consider  $W = (e_1, e_2, \dots, e_k)$  and we denote  $D[e_i][e_j]$  its distance matrix between each pair of points.

The notation  $\{p_1, p_2, \dots, p_k\}$  formalizes a representation of a partition where  $e_i$  belongs to the part  $p_i$ . The dimension of the partition is by definition given by  $dim = \max_i (p_i)$ .

---

**Algorithm 1: MINIMUMRESOLVINGPARTITION**


---

**Input:**  $D[[]][[]]$  : a distance matrix of  $n$  points  
**Output:** *Partition* : a resolving partition

```

1 for  $dim \leftarrow 2$  to  $m$  do
2    $Partition \leftarrow \text{FIRSTPARTITIONOFSIZE}(dim)$ 
3   if  $\text{ISRESOLVINGPARTITION}(Partition)$  then
4     return  $Partition$ 
5   while  $Partition \leftarrow \text{NEXTPARTITIONOFSIZE}(dim)$  do
6     if  $\text{ISRESOLVINGPARTITION}(Partition)$  then
7       return  $Partition$ 

```

---

The first algorithm that we give is simple and natural. Then we show several properties, helping to give an improved version with lower complexity.

### 3.1 Natural Method

Our aim is to determine if for a given partition, all points of  $W$  have different coordinates. We first compute all of the coordinates with the following algorithm:

---

**Algorithm 2: GETCOORDLIST** computes the list of coordinate vectors

---

**Output:**  $Coord[[]][[]]$  : the list of coordinate vectors  
**Input:**  $D[[]][[]]$  : the distance matrix ;  $Partition = \{p_1, p_2, \dots, p_m\}$  : the partition

```

1 Initialization.
2  $Coord[[]][[]] \leftarrow \infty$ 
3 Computing the coordinate vectors.
4 foreach  $point\_A \in W$  do
5   foreach  $point\_B \in W$  do
6      $part\_B \leftarrow Partition[point\_B]$ 
7     if  $Coord[point\_A][part\_B] > D[point\_A][point\_B]$  then
8        $Coord[point\_A][part\_B] \leftarrow D[point\_A][point\_B]$ 
9 return  $Coord$ 

```

---

Each coordinate of a point  $q$  is given by the minimum distance from  $q$  to points of the corresponding part. Every points belong to a part, so we need to compute the distance between each couple of points. The complexity is  $O(n^2)$ , where  $n$  is the number of points.

Now we need to know if all points are distinguished by coordinates. The answer is given by using a sorting algorithm that stops and returns *false* if two vectors are identical, but returns *true* if it ends sorting. Logically, the complexity of this step is in  $O(\text{dim} * n \log_2(n))$  comparisons of  $n$  vectors of size  $\text{dim}$ .

Finally this method requires  $O(n^2 + \text{dim} * n * \log_2(n))$  to determine if the partition is resolving.

### 3.2 Computing Coordinate Vectors in Efficient Time

**Lemma 1.** *Only two coordinates may change for each point from a partition to an adjacent one.*

*Proof.* Coordinates are given by a distance to a part (a set of points), so if this part does not change, the coordinate remains. Adjacent partitions are either in the same class with a distance of 2, which means that two points exchanged their part number, or partitions are in different classes with a distance 1, meaning that a single point has changed its part number. In both cases, two parts have changed, so two coordinates may change for any given point.  $\square$

**Lemma 2.** *Computing coordinate vectors for a given partition, knowing the coordinate vectors of an adjacent partition, can be made in  $O(n \log_2(n))$  time.*

In order to prove this lemma, we explain the algorithm and show its complexity. We use a data structure to maintain up-to-date some information for a partition. Each point needs to know an ordered list by distance to each point of each part. The more convenient structure is an indexed binary heap.

A heap is a classical data structure in computer science. This structure is based on the *heap property* for a binary tree: if the node  $A$  is a parent of the node  $B$  then we have  $\text{key}(A) < \text{key}(B)$ .

A binary heap has also the property to be always compact, i.e. all the leaves of the tree have their depths different by at most 1. In an  $n$  elements heap, the depth is  $\log_2 n$ .

Operations like FINDMIN, in constant time  $O(1)$ , DELETEMIN and INSERT, in  $\log_2 n$  time, are classically defined for min-heap. We need to define two new operations: DELETERANDOM and CHANGEKEY.

The first one is provided by an index for points in a heap. We can directly find the location of a key in a heap in constant time  $O(1)$ . Without the index, finding a random key in a heap may take  $O(n)$  operations.

CHANGEKEY can be decomposed in two operations: INCREASEKEY or DECREASEKEY. Decreasing a key in a min-heap is done by switching a node by its father while the father's key is greater than the node's one. INCREASEKEY is the reverse operation: it consists in exchanging the node by its smaller son until the son's key is greater or equal to its own key.

DELETERANDOM, which removes a random node  $r$ , can be easily defined knowing CHANGEKEY, in two steps. First, we switch the node  $r$  with the last node  $l$  of the heap, knowing that the last node is always removable. Second, we consider that the kept node  $l$  has just changed its key.



These two operations are made in the worst case in  $O(\log_2(n))$  time.

For each point of  $W$ , we keep up-to-date *dim* min-heaps, one for each parts of *Partition*. The coordinate vectors are given through a series of FINDMIN.

There exist two kinds of adjacent partitions. The first one is two partitions of the same class with distance of 2, i.e. two points from the first partition switch their parts number to become the second partition. The second kind is constituted by partitions having distance of 1, i.e. going from the first one to the second one is simply made by picking a point from a part and moving it to an other part.

Considering adjacent partitions of the first kind, the operation can be assimilated to changing a key of a point. Indeed, removing a point from a heap, then directly adding an other point is equivalent to consider that the key of the leaving point is changed by the key of the incoming point.

This operation requires two calls of CHANGEKEY for each point, so its complexity can be approximated by  $O(n \log_2(n))$  operations. This is an upper bound because the number of points contained in two heaps is always smaller than  $n$  since each point belongs to one and only one heap.

Considering now the second kind of adjacent partitions, the update operations are made by a call of DELETERANDOM on the heap containing the leaving node, and a call of INSERT into an other part.

The complexity of this update is in  $O(n \log_2(n))$ , which is an upper bound for the same reasons explained just before.

### 3.3 Comparing Coordinate Vectors in Efficient Time

We now show how to improve the step of determining if two coordinate vectors are identical.

**Lemma 3.** *Two points of distinct parts of a partition are always distinguished.*

*Proof.* Every points of a part are at zero distance from it, while outer points have positive distance from it. Thus coordinates are different.  $\square$

Thanks to the lemma 3, we could improve the second part of the sorting algorithm, described in section 3.1. Indeed, the sorting step can be split into *dim* independent subproblems.

**Lemma 4.** *Comparing coordinate vectors for a given partition knowing the coordinate vectors can be made in optimal time  $O(n * dim)$  (the size of the coordinate vectors).*

All of the coordinates are bounded by  $[0, d_{max}]$ , where  $d_{max}$  is the diameter of  $W$ . Since we restrict to discrete distances, which have integer values, the maximum number of distinct distances is  $d_{max}$ . This number is lower than  $n$  for rectangles in  $\mathbb{Z}^k$ , due to the fact that  $\mathbb{Z}^k$  is a module (a discrete vector space) and the maximum number of distinct vectors (up to a symmetry) is bounded by the number of vectors from a corner. This is also true in general case (not

a rectangle) for distances  $d_1$  and  $d_\infty$ , for which  $d_{max}$  is exactly the number of different possible distance values.

Considering this, we can use a specific data structure: a *trie*. A trie (or *prefix tree*) is an ordered tree data structure commonly used to store strings. In this specific structure, unlike in a binary tree, a node is not associated to a specific key but the position of the node in the trie determines a specific prefix of the key.

In our case, we define a node as a structure containing two fields: a table of size  $d_{max}$  pointers of nodes, and an integer which indicates the current number of nodes contained in the table. This allows us to create a table containing all the coordinate vectors where the depth  $i$  in the trie corresponds to the coordinate  $i$  in a vector. Each leaf of the created trie will have a depth of  $dim$ , considering that the coordinates are given by the parts of a partition.

This data structure provides INSERTKEY, DELETEKEY and FINDKEY in  $O(m)$  where  $m$  is the size of the key. Finally, we can search in optimal time for identical coordinate vectors, because we only consider once each vector and each of its coordinates.

### 3.4 Complexity

We just described structures and procedures to improve the research of a resolving partition using properties of local changes; we now study the complexity in time and space.

We have previously seen that each point needs a min-heap indexed structure of size  $2n$  for each of the  $dim$  part, and finally a trie composed by nodes of size  $n$ . A larger upper bound of the number of nodes is given by the number of points  $n$  multiplied by the depth  $dim$  in the trie and the size of a node. We consider  $sizeof(node) \approx n$  because we are interested in discrete geometry problems and then we can use an index table for distances in order to use less memory as possible by compacting the table pointer of each node.

So the complexity in space is  $O(n * dim + n^2 * dim)$ .

Analyzing the complexity in time is done in two steps: analyze the initialization, then analyze the looping steps (given by the described procedures).

For recall, a looping step is composed in two stages: updating structures in order to change the coordinate vectors from a partition to an adjacent one; then updating the trie in order to search undistinguished points. For each of the  $n$  points, 2 min-heap with combined size less than  $n$  will be updated. Updating the trie is in the worst case: changing every of the  $n$  key of size  $dim$ , by removing a key and adding a new one.

So the complexity in time for local changes is  $O(n \log_2 n + n * dim)$ .

Time complexity for the initialization is bigger because it is not a local change, indeed every point will be attributed to a part and all parts will change.

In the worst case every points are added to the same part, so for each point, we will create a min-heap of size  $n$  by successively pushing each point. This is done in  $O(n \log_2 n)$ . Once made, the coordinate vectors are added to the trie, so time complexity for initialization is  $O(n^2 \log_2 n + n * dim)$ .

## 4 Dimension for Gauges

We have seen in previous section that searching partition dimension by enumeration is actually time consuming. Enumerating is not mandatory for simple cases; we establish in this section the resolving partitions for common distances in rectangles and convex shapes.

We briefly recall the definition of a gauge. Given a convex  $\mathcal{C}$  containing the origin  $O$  in its interior, a *gauge* for  $\mathcal{C}$  is the function  $\gamma_{\mathcal{C}}(x)$  defined by the minimum positive scale factor  $\lambda$ , necessary for having  $x \in \lambda\mathcal{C}$ . Formally,  $\gamma_{\mathcal{C}}(x) = \inf \{ \lambda \in \mathbb{R}_+ : x \in \lambda\mathcal{C} \}$ . When  $\mathcal{C}$  is a polyhedron,  $\gamma_{\mathcal{C}}$  is called a polyhedral gauge. By definition, all norms are gauges for their unit ball. Conversely, a gauge for  $\mathcal{C}$  is a norm, denoted  $n_{\mathcal{C}}$ , iff  $\mathcal{C}$  is central-symmetric. We call *distance gauge*, denoted  $d_{\mathcal{C}}$ , the metric induced by a central-symmetric gauge  $\gamma_{\mathcal{C}}$ . We only consider central-symmetric gauges in the paper.

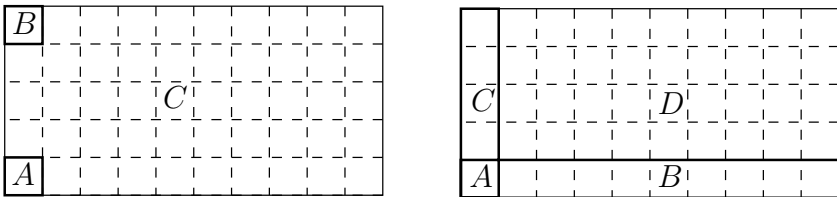
### 4.1 Dimension in Rectangles

**Lemma 5.** *The partition dimension of gauges in a rectangle is 3 or 4.*

We have shown in [12] that the metric dimension for gauges is always 2 in a rectangle except for the polyhedral gauges which have a vertical or horizontal facet in their ball. Hence, the partition dimension is at most  $2+1$  for that kind of gauges. For the others gauges we can provide a simple pattern which always gives a resolving 4-partition (see figure 1): the first part is the bottom-left corner of the considered rectangle; the second part is the remaining points of the bottom line; the third part is the remaining points of the first column; the fourth part is constituted by all other points.

Note that the coordinates given by the fourth part are useless for the resolving problem, but are necessary to get an actual partition of the rectangle.

The lemma is still valid for non-gauge distances whose balls are all convex.



**Fig. 1.** Illustration of the lemma 5. On the left a resolving 3-partition of the rectangle for gauges which do not contain vertical nor horizontal facet built using metric basis pattern (see [12]). On the right, the pattern used to prove that any convex distance has at least a resolving 4-partition in a rectangle.

## 4.2 Dimension in Convex Sets

Tomescu studied partition dimension in space  $\mathbb{Z}^2$  for  $d_1$  and  $d_\infty$ . Here we extend that study to every gauges, and to another unusual family of distances: the non-homogeneous distances whose every balls are convex.

**Lemma 6.** *In the digital plane  $\mathbb{Z}^2$ , the partition dimension for any gauge which has no vertical nor horizontal facet is 3.*

A proof of this lemma is given using the same pattern proposed by Tomescu: a perpendicular split of the plane in three areas, as follows.  $A$  is composed by points verifying ( $y < 0$ );  $B$  is the set of points corresponding to ( $y \geq 0$  and  $x < 0$ ); finally  $C$  is the set of the remaining points ( $y \geq 0$  and  $x \geq 0$ ). Using lemma 3 we need to prove that the points inside an area are distinguished (see figure 2).

- Points in  $B$  are distinguished: considering the line  $y = -1$  as a centre of a distance ball which gives the coordinate distance from the part  $A$  to the part  $B$ , the frontier of the ball consists in a vertical line moving through  $B$ . Distance from the  $C$  area are given by an horizontal line parallel to the intersection between  $B$  and  $C$ . Intersection of both of the balls line results in a single point, so each point in  $B$  has unique coordinates.
- Points in  $C$  are distinguished: for the exact same reasons explained for the  $B$  area ( $B$  and  $C$  play symmetric roles), every point in  $C$  has unique coordinates.
- Points in  $A$  are distinguished: in order to prove this, we look at the upper part of  $A$  where points are closer to the  $C$  area. In this part, the front line of the  $C$  ball consists in a vertical line, but the front line of the  $B$  ball is a strictly monotonic curve which does not have any vertical nor horizontal part because we use gauges with these properties, and so intersection between both of this front line results in a single point. The same result happens in the lower part of  $A$  by symmetries of the  $B$  and  $C$  part.  $\square$

**Lemma 7.** *In the digital plane  $\mathbb{Z}^2$ , the partition dimension for any convex distance is lower than 4.*

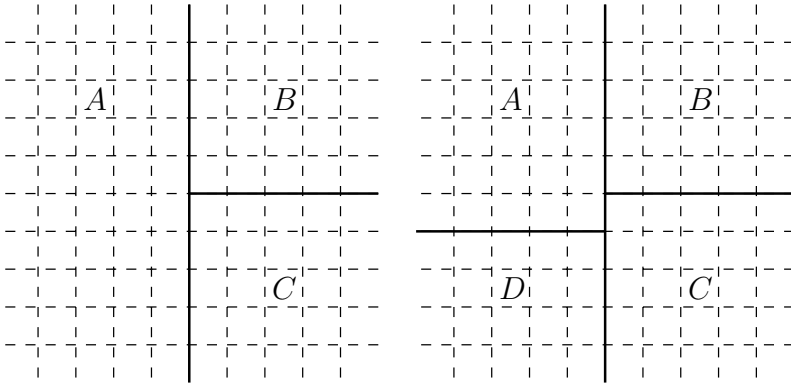
To prove that we also use the same pattern that Tomescu used to prove the dimension of  $d_\infty$  in  $\mathbb{Z}^2$ . It is nearly the same pattern used for demonstrating the previous lemma using an additional area  $D$  which takes off the  $A$  area of its lower points ( $y < -1$ ), see figure 2.

It is clear that in this configuration, points in the areas  $C$  and  $D$  have unique coordinates for the same reasons explained to prove uniqueness properties in the area  $B$  and  $C$  in the previous lemma.

Both of the previous lemma 6 and 7 remain valid in the continuous space  $\mathbb{R}^2$ , but a stronger property exists: indeed we have

**Lemma 8.** *In the continuous space  $\mathbb{R}^2$ , the partition dimension for  $d_\infty$  is 3.*

*Proof.* Applying a  $45^\circ$  rotation to the perpendicular pattern, we get the exact same case presented for  $\dim(d_1, \mathbb{Z}^2)$  (up to the rotation).  $\square$



**Fig. 2.** On the left, illustration of the pattern used to prove the lemma 6. This always gives a resolving 3-partition of the space  $\mathbb{Z}^2$  for gauges which do not contain vertical nor horizontal facet. On the right, the pattern used to prove the lemma 7. That pattern gives a resolving 4-partition for any convex distance in  $\mathbb{Z}^2$ .

Then it is clear that

**Corollary 1.** *For any distance  $d$ ,  $\dim(d, \mathbb{Z}^2) \geq \dim(d, \mathbb{R}^2)$ .*

Considering lemma 8, we finally conjecture that

*Conjecture 1.* For any convex distance  $d$ , we have  $\dim(d, \mathbb{R}^2) = 3$ .

## 5 Conclusion

In this paper, we have studied the partition dimension problem in the Euclidean and discrete spaces, for usual distance functions used in digital geometry. We have established combinatorial results for the problem, then we have proposed algorithms verifying the resolving property of a partition and stated their complexity. We have finally presented resolving partitions and bounds for geometrical cases.

The procedures presented in the algorithmic section are applied for computing an enumeration algorithm. They could also be applied for a randomized local search algorithm, because the adjacency notion is exactly the notion of locality required by that kind of algorithm.

Unlike for the metric basis, the partition dimension seems to be only related to the distance, while being independent from the considered convex  $W$ . The combinatorics of the number of possible partitions explodes with the size of  $W$ ; notwithstanding, the partition dimension is bounded by a low number, which makes the enumeration achievable.

## References

1. Buczkowski, P., Chartrand, G., Poisson, C., Zhang, P.: On  $k$ -dimensional graphs and their bases. *Periodica Mathematica Hungarica* 46, 9–15 (2003)
2. Cáceres, J., Hernando, C., Mora, M., Pelayo, I., Puertas, M.: On the metric dimension of infinite graphs. *Electronic Notes in Discrete Math.* 35, 15–20 (2009)
3. Chappell, G.G., Gimbel, J.G., Hartman, C.: Bounds on the metric and partition dimensions of a graph. *Ars Comb.* 88, 349–366 (2008)
4. Chartrand, G., Eroh, L., Johnson, M., Oellermann, O.: Resolvability in graphs and the metric dimension of a graph. *Discrete Applied Math.* 105(1-3), 99–113 (2000)
5. Chartrand, G., Salehi, E., Zhang, P.: On the partition dimension of a graph. *Congressus Numerantium* 131, 55–66 (1998)
6. Chartrand, G., Salehi, E., Zhang, P.: The partition dimension of a graph. *Aequationes Mathematicae* 59, 45–54 (2000)
7. Harary, F., Melter, R.: On the metric dimension of a graph. *Ars Combinatoria* 2, 191–195 (1976)
8. Hardy, G., Wright, E.: An introduction to the theory of numbers, 5th edn. Oxford University Press (October 1978)
9. Hernando, C., Mora, M., Pelayo, I., Seara, C., Cáceres, J., Puertas, M.: On the metric dimension of some families of graphs. *Electronic Notes in Discrete Mathematics* 22, 129–133 (2005)
10. Khuller, S., Raghavachari, B., Rosenfeld, A.: Landmarks in graphs. *Discrete Applied Mathematics* 70(3), 217–229 (1996)
11. Melter, R., Tomescu, I.: Metric bases in digital geometry. *Computer Vision, Graphics, and Image Processing* 25(1), 113–121 (1984)
12. Rebatel, F., Thiel, É.: Metric Bases for Polyhedral Gauges. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) *DGCI 2011*. LNCS, vol. 6607, pp. 116–128. Springer, Heidelberg (2011)
13. Tomescu, I.: Discrepancies between metric dimension and partition dimension of a connected graph. *Discrete Mathematics* 308(22), 5026–5031 (2008)

# Walking in the Farey Fan to Compute the Characteristics of a Discrete Straight Line Subsegment

Isabelle Sivignon

`gipsa-lab`, CNRS, UMR 5216, F-38420, France  
`isabelle.sivignon@gipsa-lab.grenoble-inp.fr`

**Abstract.** Given a Digital Straight Line (DSL) of known characteristics  $(a, b, \mu)$ , we address the problem of computing the characteristics of any of its subsegments. We propose a new algorithm as a smart walk in the so called Farey Fan. We take profit of the fact that the Farey Fan of order  $n$  represents in a certain way all the digital segments of length  $n$ . The computation of the characteristics of a DSL subsegment is then equivalent to the localization of a point in the Farey Fan. Using fine arithmetical properties of the fan, we design a fast algorithm of theoretical complexity  $\mathcal{O}(\log(n))$  where  $n$  is the length of the subsegment. Experiments show that our algorithm is faster than the one previously proposed by Said and Lachaud in [15,14] for “short” segments.

**Keywords:** Digital geometry, Digital straight segment recognition, Farey fan.

## 1 Introduction

Digital Straight Lines (DSL) and Digital Straight Segments (DSS) have been known for many years to be interesting tools for digital curve and shape analysis. The applications range from simple coding to complex multiresolution analysis and geometric estimators. All these applications require to solve the so-called DSS recognition problem. Many algorithms, using arithmetics, combinatorics or dual-space have been proposed to solve this problem, reaching a computational complexity of  $\mathcal{O}(n)$  for a DSS of length  $n$ . A DSS belongs to infinitely many DSL of different characteristics, only one DSL enables to define the minimal characteristics of a DSS. In [14], the authors introduce the following problem: given a DSL of known characteristics and a subsegment of this DSL, compute the minimal characteristics of the DSS. The authors originally encountered this problem for implementing a fast algorithm to compute a multiresolution representation of a contour. This problem also arises for the digitization of a segment given by its two floating-point endpoints: indeed, the slope computed from the endpoints may be quite far from the minimal characteristics of the digitized segment, especially if the segment is short. Two algorithms (SmartDSS and ReversedSmartDSS) are presented in [14,15]: both use the decomposition into continuous fractions of the DSL slope and reach a logarithmic complexity.

This problem is however not so new since in [12], the author presents a quick sketch of a method that solves it using the Farey Fan. The announced complexity of the method is  $\mathcal{O}(\log^2 n)$  for a segment of length  $n$ . In this paper, we investigate further in this direction to provide a thoroughly defined algorithm. Moreover, we show how its complexity can be lowered to  $\mathcal{O}(\log(n))$  with an astute use of arithmetical properties of the Farey Fan. Finally, we compare the performance of our algorithm with the ones proposed in [14] and [15] and show that it behaves particularly well for “short” segments.

## 2 Setting the Problem

### 2.1 Digital line, Segment and Minimal Characteristics

A *Digital Straight Line* (DSL for short) of integer characteristics  $(a, b, \mu)$  is the infinite set of digital points  $(x, y) \in \mathbb{Z}^2$  such that  $0 \leq ax - by + \mu < \max(|a|, |b|)$  ( $\gcd(a, b) = 1$ ) [5]. These DSL are 8-connected and often called *naive*. The slope of the DSL is the fraction  $\frac{a}{b}$  and  $\frac{\mu}{b}$  is the shift at the origin. In the following, without loss of generality, we assume that  $0 \leq a \leq b$ . The remainder of a DSL of characteristics  $(a, b, \mu)$  for a given digital point  $(x, y)$  is the value  $ax - by + \mu$ . The *upper (resp. lower) leaning line* of a DSL is the straight line  $ax - by + \mu = 0$  (resp.  $ax - by + \mu = b - 1$ ). Upper (resp. lower) leaning points are the digital points of the DSL lying on the upper (resp. lower) leaning lines.

A *Digital Straight Segment* (DSS) is a finite 8-connected part of a DSL. It can be uniquely defined by the characteristics of a DSL containing it and two endpoints  $P$  and  $Q$ . However, a DSS belongs to an infinite number of DSLs. In this context, the *minimal characteristics* of a DSS are the characteristics of the DSL containing it with minimal  $b$  [16]. Note that the notions of leaning points and lines are similarly defined for DSSs. DSS recognition algorithms aim at computing the minimal characteristics of a DSS, taking profit of the following fact:  $(a, b, \mu)$  are the minimal characteristics of a DSS if and only if the DSS contains at least three leaning points [5]. In this case, the minimal characteristics are the characteristics of the DSS upper leaning line.

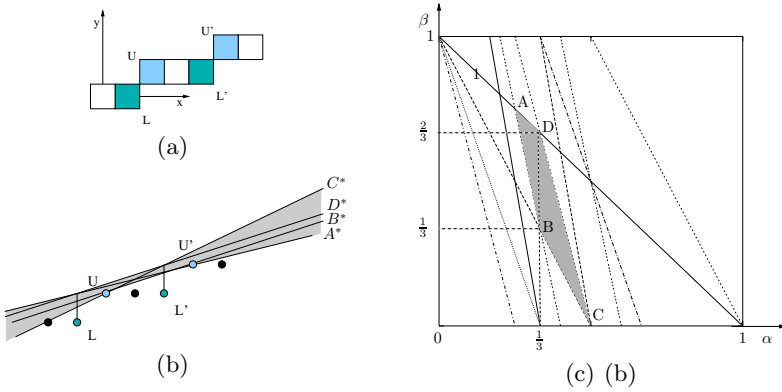
The set of DSLs containing a DSS is usually called the *preimage* of the DSS. Given a DSS  $S$ , it is defined as  $\mathcal{P}(S) = \{(\alpha, \beta), |\alpha| \leq 1 \mid \forall (x, y) \in S, 0 \leq \alpha x - y + \beta < 1\}$ . The preimage can be represented in the  $(\alpha, \beta)$  space, where  $\alpha$  represents the slope and  $\beta$  the shift at the origin of a straight line.

The preimage of a DSS is a polygon with a well-defined structure that is directly related to the leaning points and lines defined by its minimal characteristics [12,6]. Figure 1 below (from [4]) illustrates this point.

**Proposition 1** ([4]). *Let  $\mathcal{P}(S)$  be the preimage of  $S$ . Let  $ABCD$  be the polygon defined by this preimage, where  $A$  is the upper left most vertex, and the vertices are named counterclockwise. Following the notations of Figure 1 we have:*

- The vertex  $B$  maps to the upper leaning line  $UU'$ ;
- The vertex  $D$  maps to the lower leaning line  $LL'$  translated by the vector  $(0, 1)$  in the digital space;





**Fig. 1.** (a) DSS of minimal characteristics  $(1, 3, 1)$  with its leaning points  $U, U', L, L'$ . (c) Representation of  $\mathcal{P}(S)$  in the  $(\alpha, \beta)$  space. (b) Each vertex of the preimage maps to a straight line in the digital space. The vertex  $B(\frac{1}{3}, \frac{1}{3})$  maps to the upper leaning line, the characteristics of which are the minimal characteristics of the DSS.

- The vertex  $A$  maps to the straight line  $U'L^+$ , where  $L^+ = L + (0, 1)$ ;
- The vertex  $C$  maps to the straight line  $UL'^+$ , where  $L'^+ = L' + (0, 1)$ .

The minimal characteristics of  $S$  are  $(a, b, \mu)$  if and only if  $B = (\frac{a}{b}, \frac{\mu}{b})$  ( $p$  and  $q$  relatively prime).  $B$  is called the **characteristic point** of  $\mathcal{P}(S)$ . Edges  $[AB]$  and  $[BC]$  are called **lower edges**.

## 2.2 Farey Fan

**Definition 1 (Ray).** Let  $x$  and  $y$  be two nonnegative integers. The ray defined by  $x$  and  $y$  is defined and denoted as follows:

$$R(x, y) = \{(\alpha, \beta) | \beta = -x\alpha + y\}$$

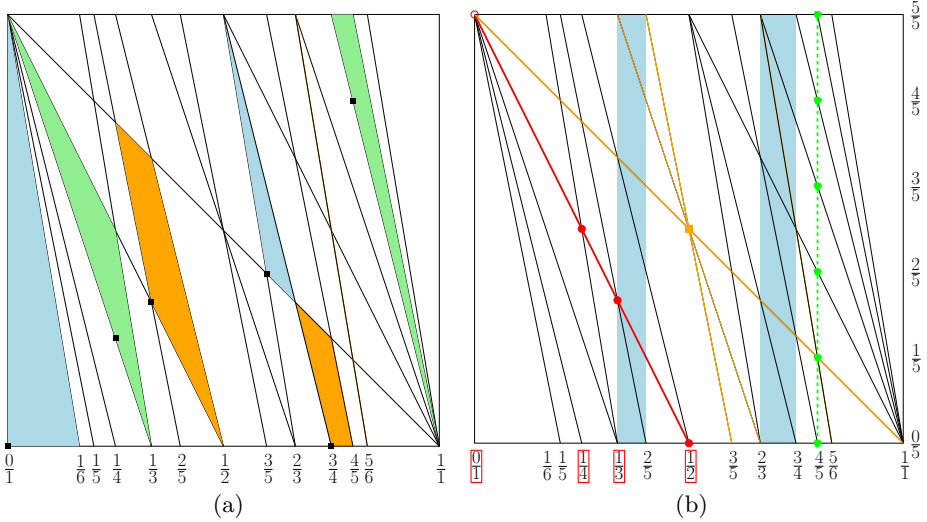
The slope of the ray is  $x$ .

Note that  $x$  is not the geometrical slope of the ray but its absolute value. In the following, the order on the slopes is to be understood as the order on the absolute values of the geometrical slopes.

**Definition 2 (Farey Fan).** The Farey Fan of order  $n$ , denoted by  $\mathcal{F}_n$  is defined in the  $(\alpha, \beta)$  space as the arrangement of all the rays  $R(x, y)$  such that  $0 \leq y \leq x \leq n$ , and such that  $0 \leq \alpha \leq 1$  and  $0 \leq \beta \leq 1$ .

A facet of  $\mathcal{F}_n$  is a cell of dimension 2 of this arrangement. In the following, a point of  $\mathcal{F}_n$  stands for any point  $v$  of the  $(\alpha, \beta)$  space ( $0 \leq \alpha \leq 1$  and  $0 \leq \beta \leq 1$ ) belonging to a ray, and such that the abscissa of  $v$  is a fraction of denominator smaller than or equal to  $n$ .

For any  $n$ , it is well known that there is a bijection between the facets of  $\mathcal{F}_n$  and the set of DSSs of length  $n$  (composed of  $n + 1$  pixels) [12].



**Fig. 2.** (a) Farey Fan of order 6. (b) Illustration of Properties 1 to 4 from Section 3.

**Definition 3.** Let  $S$  be a DSS of length  $n$ .  $\text{Facet}(S)$  is the facet equal to  $\mathcal{P}(S)$  in the Farey fan of order  $n$ .

Moreover, from Proposition 1, a one-to-one correspondence can be defined between a facet and the characteristic point of the facet.

**Definition 4.** Let  $f$  be a facet of the Farey fan of order  $n$ . We denote by  $\text{CPoint}(f)$  the point  $v$  of  $f$  such that, if  $v = (\frac{p}{q}, \frac{r}{q})$ , then  $(p, q, r)$  are the minimal characteristics of the DSS  $\text{Facet}^{-1}(\text{CPoint}^{-1}(v))$ .

The Farey Fan of order 6 is depicted in Figure 2(a). The characteristic points of a few facets are depicted. Note that three types of facets can be identified:

- quadrilateral facets (in orange in Figure 2(a));
- upper triangular facets (in green in Figure 2(a));
- lower triangular facets (in blue in Figure 2(a)).

Consider now the following problem:

*Problem 1.* Given a DSL  $L$  of characteristics  $(a', b', \mu')$  and two points  $P(x_P, y_P)$  and  $Q(x_Q, y_Q)$  of this DSL, compute the minimal characteristics  $(a, b, \mu)$  of the DSS  $S = \{(x, y) \in L \mid x_P \leq x \leq x_Q\}$ .

After a translation of the characteristics of  $L$  such that  $P$  is set to the origin ( $\mu \leftarrow \mu + ax_P - by_P$ ), this problem is equivalent to the following one:

*Problem 2.* Given a point  $\mathbf{\Lambda}(\frac{a}{b}, \frac{\mu}{b})$  and a point  $Q(x_Q, y_Q)$ , find the point  $v$  of the Farey fan of order  $n = x_Q$  such that  $\mathbf{\Lambda} \in \text{CPoint}^{-1}(v)$ .

In other words, the problem is to find the characteristic point of the facet of  $\mathcal{F}_n$  containing  $\mathbf{\Lambda}$ .

All in all solving Problem 2 is equivalent to performing a point location in an arrangement of lines. However, the number of facets in the Farey fan of order  $n$  (which is equal to the number of DSS of length  $n$ ) is in  $\mathcal{O}(n^3)$  [9,10,2], and point location algorithms in such a structure are expensive in term of both time and space complexity [13]. This brute force approach is then less efficient than classical DSS recognition algorithms [5,17,11,7].

In the following sections, we revisit the approach proposed by [12] and present an algorithm to solve Problem 2 in time complexity  $\mathcal{O}(\log n)$ , without explicitly computing the Farey fan. In the next section, we recall several structural and arithmetical properties of the Farey fan, and derive some very useful corollaries. These properties are the core of the algorithm detailed in section 4.

### 3 Properties of the Farey Fan

The Farey series of order  $n$  is the set of irreducible fractions in  $[0, 1]$  of denominator lower than or equal to  $n$  [8]. All the properties below are illustrated in Figure 2(b) in the Farey fan of order 6. The first three properties are from [12] and the reader is invited to consult this reference for the proofs, that are fairly simple.

*Property 1 ([12]).* The abscissas of intersections of a ray  $R(x, y)$  of  $\mathcal{F}_n$  with other rays are **consecutive terms** of a Farey series of order  $\max(x, n - x)$ .

In Figure 2(b), the abscissas of the intersections between the ray  $R(2, 1)$ , depicted in red, and the other rays of  $\mathcal{F}_6$  are consecutive terms of the Farey series of order  $4 = \max(2, 6 - 2)$ .

*Property 2 ([12]).* Let  $f_i$  and  $f_{i+1}$  be two consecutive fractions of the Farey series of order  $n$ . In the interval  $f_i < \alpha < f_{i+1}$ , there is no intersection of rays. Thus, in this interval the Farey fan is a simple **ladder** of rungs.

In Figure 2(b), two ladders are depicted in blue for  $f_i = \frac{1}{3}$  and  $f_i = \frac{2}{3}$ .

*Property 3 ([12]).* Let  $v(\frac{p}{q}, \frac{r}{q})$ ,  $0 \leq p \leq q \leq n$ , be a point of  $\mathcal{F}_n$ . Let  $R(x_0, y_0)$  be the ray of minimum slope passing through  $v$ . The other rays passing through  $v$  have a slope equal to  $x_0 + kq$  with  $k \in \mathbb{Z}$  and  $x_0 + kq \leq n$ .

In Figure 2(b), three rays go through the point  $(\frac{1}{2}, \frac{1}{2})$  (in orange). The slopes of these rays are equal to  $x_0 = 1, 3$  and  $5$ . From this property, we can derive the following corollary.

**Corollary 1.** *Let  $v(\frac{p}{q}, \frac{r}{q})$ ,  $0 \leq p \leq q \leq n$ , be a point of  $\mathcal{F}_n$ . Let  $R(x, y)$  be a ray passing through  $p$ .  $\bar{R}$  is the ray of smallest slope passing through  $v$  if and only if  $x - q < 0$ . It is the ray of greatest slope passing through  $v$  if and only if  $x + q > n$ .*

*Property 4.* Let  $\frac{p}{q}$  be a fraction of the Farey series of order  $n$ . The intersection between the line  $\alpha = \frac{p}{q}$  and  $\mathcal{F}_n$  is exactly the set of points  $(\frac{p}{q}, \frac{r}{q})$  where  $r$  takes all the integer values between 0 and  $q$ .

*Proof.* We study the intersection between  $R(x, y)$  defined by the equation  $\beta = -\alpha x + y$  and  $\alpha = \frac{p}{q}$ . We get  $\beta = \frac{-px+qy}{q}$ . For  $0 \leq y \leq x \leq q \leq n$ , the quantity  $-px + qy$  takes all the integral values in the interval  $[[0, q]]$ , which ends the proof.

In Figure 2(b), the intersection between  $\alpha = \frac{4}{5}$  (depicted in green) and  $\mathcal{F}_n$  is the set of points  $(\frac{4}{5}, \frac{r}{5})$  with  $r \in \mathbb{Z}$ ,  $0 \leq r \leq 5$ . Using Properties 2 and 4, we can prove the following result to compute the ray of smallest slope in a given point.

**Corollary 2.** *Let  $v(\frac{p}{q}, \frac{r}{q})$ ,  $0 \leq p \leq q \leq n$ , be a point of  $\mathcal{F}_n$ . Let  $\frac{p'}{q'}$  be the fraction following  $\frac{p}{q}$  in the Farey Series of order  $n$ . The ray of smallest slope passing through  $v$  is defined by the point  $v$  and the point of coordinates  $v'(\frac{p'}{q'}, \frac{\lfloor \frac{r q'}{q} \rfloor}{q'})$ .*

*Proof.* From Property 2,  $\mathcal{F}_n$  is a ladder in the interval  $[\frac{p}{q}, \frac{p'}{q'}]$ , which means there is no intersection of rays in this interval. From Property 4, we know that there is at least one ray passing through the point  $v$ . Again from Property 4, all the rays passing through  $v$  cut the line of equation  $\alpha = \frac{p'}{q'}$  in a point  $v'(\frac{p'}{q'}, \frac{r'}{q'})$ ,  $r' \in \mathbb{Z}$ ,  $0 \leq r' \leq q'$ . Among all these rays, the ray of smallest slope is the one that passes through the point  $v_{max}(\frac{p'}{q'}, \frac{r_{max}}{q'})$  where  $r_{max}$  is the maximal value of  $r'$  such that  $\frac{r'}{q'} \leq \frac{r}{q}$ .  $r_{max}$  is given by  $\lfloor \frac{r q'}{q} \rfloor$  which ends the proof.

## 4 Fast Walk in the Farey Fan

Following Problem 2, we look for the characteristic point of the facet of  $\mathcal{F}(n)$  containing a given point  $\mathbf{\Lambda}(\frac{a}{b}, \frac{r}{b})$ . From Proposition 1, Section 2.2 and Property 4 we have the following characterization of the characteristic point.

*Property 5.* A point  $v(\frac{p_v}{q_v}, \frac{r_v}{q_v})$  is the characteristic point of a facet if and only if:

1. either  $v$  is the intersection of the two lower edges:
  - (a) the ray supporting the right lower edge is the one of smallest slope in  $v$ ;
  - (b) the ray supporting the left lower edge is the one of greatest slope in  $v$ ;
2. or  $v$  is on the unique lower edge and more than one ray passes through the point  $(\frac{p_v}{q_v}, \frac{r_v+1}{q_v})$

As in [12], the algorithm consists of three steps that are detailed in the following sections:

1. Find the ladder to which  $\mathbf{\Lambda}$  belongs;
2. Locate the highest ray that lies on or below  $\mathbf{\Lambda}$ : this ray supports a lower edge of the facet (Section 4.2, Algorithm 1);
3. Walk along the ray(s) to determine the characteristic point (Section 4.3, Algorithm 2).

Particular cases where  $\mathbf{\Lambda}$  is a point of  $\mathcal{F}_n$  (either on a ray, or a vertex) are eluded, so that the focus is done on the general case. However, these particular cases are not complicated to handle.

#### 4.1 Find the Ladder

Given a point  $\mathbf{\Lambda}(\frac{a}{b}, \frac{\mu}{b})$ , finding the ladder to which  $\mathbf{\Lambda}$  belongs in  $\mathcal{F}_n$  is equivalent to finding the two fractions with a denominator smaller than  $n$  closest to  $\frac{a}{b}$  (greater and lower). We look for two fractions  $f = \frac{p}{q}$  and  $g = \frac{p'}{q'}$  such that  $q \leq n$ ,  $q' \leq n$ ,  $f \leq \frac{a}{b} \leq g$ , and there is no fraction of denominator smaller or equal to  $n$  neither between  $f$  and  $\frac{a}{b}$  nor between  $\frac{a}{b}$  and  $g$ .

This problem is closely related to the computation of the best rational approximation of a number, for which solutions using the decomposition into continuous fractions exist [8]. However, we do not need only the best approximation, which is either the closest lower or closest greater fraction, but also the other one. To solve this problem, we use the algorithm of Charrier and Buzer [3]. This algorithm aims at computing the approximation of any real number by rational numbers of bounded denominator and straightforwardly solves our problem in  $\mathcal{O}(\log(n))$ . Moreover the algorithm is simple to implement and does not require continuous fractions computations.

#### 4.2 Locate a Lower Edge

At this point, we work in a ladder defined by two fractions  $f = \frac{p}{q}$  and  $g = \frac{p'}{q'}$  of  $\mathcal{F}_n$ . This step consists in localising  $\mathbf{\Lambda}$  in the ladder by computing the highest ray under  $\mathbf{\Lambda}$  in  $\mathcal{F}_n$ . In [12], this step is performed as a binary search among the rays of the ladder. However, each stage of the binary search requires to solve a diophantine equation with the extended Euclidean algorithm, reaching a total complexity of  $\mathcal{O}(\log^2 n)$ .

Our algorithm, presented in Algorithm 1 and illustrated in Figure 3, also performs a dichotomy (line 4), but only on the rays of smallest slope passing through the points of abscissa  $\frac{p}{q}$  (in red in Figure 3).

Thanks to Property 4, this set of points can be defined as on line 1, and the rays of smallest slope are computed in time  $\mathcal{O}(1)$  in the ladder using Corollary 2 (line 2). On line 4, the ray of greatest slope is computed from the ray of smallest

---

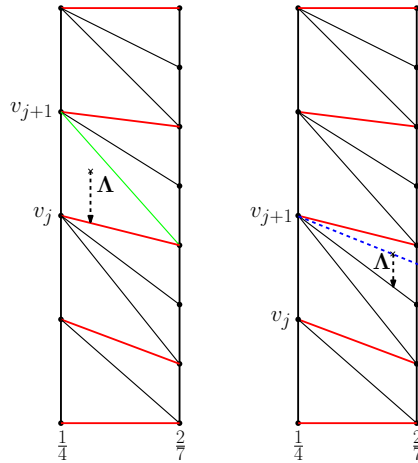
#### Algorithm 1. Search in the ladder

---

- 1 Let  $v_i = (\frac{p}{q}, \frac{i}{q})$ ,  $i \in \mathbb{Z}$ ,  $0 \leq i \leq q$  ;
  - 2 Let  $R_i(x_i, y_i)$  be the ray of smallest slope passing through  $v_i$  ;
  - 3 Perform a dichotomy on the  $R_i$  to compute  $j \in \llbracket 0, q - 1 \rrbracket$  such that  $\mathbf{\Lambda}$  is above  $R_j$  and below  $R_{j+1}$  ;
  - 4 **if**  $\mathbf{\Lambda}$  is under the ray of greatest slope through  $v_{j+1}$  **then** Return  $R_j$  ;  
**else**
  - 5     Compute the slope  $x$  of the line passing through  $v_{j+1}$  and  $\mathbf{\Lambda}$  ;
  - 6     Compute  $\lfloor x \rfloor$  as the value  $x_{j+1} + kq$  nearest to and lower than  $x$ ,  $k \in \mathbb{Z}$  ;
  - 7     Return  $R(\lfloor x \rfloor, \frac{(j+1)+p \cdot \lfloor x \rfloor}{q})$  ;
-

slope thanks to Property 3. On line 5, the value  $x$  is not an integer value, but the closest lower ray can be easily computed using Property 3.

In Figure 3, on the left, the point  $\mathbf{\Lambda}$  is located under the ray of greatest slope passing through  $v_{j+1}$  (in green, line 4 in Algorithm 1),  $R_j$  is returned. On the right, the point  $\mathbf{\Lambda}$  is in between the rays passing through  $v_{j+1}$  : the slope of the line passing through  $v_{j+1}$  and  $\mathbf{\Lambda}$  is computed (in blue, line 5 in Algorithm 1), and is rounded to find the nearest lower ray.



**Fig. 3.** Illustration of Algorithm 1: the dichotomy is performed on the red rays only

### 4.3 Find the Characteristic Point

Let us denote by  $M$  and  $N$  the two points defined as the intersection between the ray  $R(x, y)$  returned by Algorithm 1 and the vertical lines defining the ladder, i.e.  $\alpha = \frac{p}{q}$  and  $\alpha' = \frac{p'}{q'}$  as defined in Section 4.1. The segment  $[MN]$  is part of a lower edge of the facet of  $\mathcal{F}_n$  containing  $\mathbf{\Lambda}$  in  $\mathcal{F}_n$ .

The first step of the algorithm detailed in Algorithm 2 is to compute the extremities of the lower edge containing  $[MN]$ . To do so, the key point is to use Property 1 to characterize the points of intersection between a ray and other rays. Given a ray  $R(x, y)$  of the Farey Fan  $\mathcal{F}_n$  and a point  $v(\frac{p}{q}, \frac{r}{q})$  on this ray,  $v$  is the crossing point of several rays if and only if  $q \leq \max(x, n - x)$ . Thus, the abscissa of the left (resp. right) extremity of the lower edge is given by the term of the Farey series of order  $\max(x, n - x)$  lower than (resp. greater than) and closest to  $\frac{p}{q}$  (resp.  $\frac{p'}{q'}$ ) (line 1 of Algorithm 2). Given a fraction, computing the next term in a Farey series of given order cannot be solved in constant time but requires a call to the extended Euclidean algorithm. From these two fractions  $\frac{p}{q}$  and  $\frac{p'}{q'}$  we compute the two points  $\underline{Q}$  of  $R$  with abscissa equal to  $\frac{p}{q}$  and  $\overline{O}$  of  $R$  with abscissa equal to  $\frac{p'}{q'}$  (line 2).

At this point,  $[Q\bar{O}]$  is a lower edge of the facet containing  $\Lambda$ . Then, the three cases illustrated in Figure 4 can occur: either  $\underline{Q}$  or  $\bar{O}$  is the characteristic point (case (a) and (b)), or not (case (c)). We use Property 5 to distinguish between these cases:

- if  $R$  is the ray of smallest slope in  $\underline{Q}$ , then  $\underline{Q}$  is the characteristic point: the condition line 3 refers to Corollary 1;
- if  $R$  is the ray of greatest slope in  $\bar{O}$ , then  $\bar{O}$  is the characteristic point: the condition line 4 refers to Corollary 1;
- otherwise, the facet is lower triangular, and the abscissa of the characteristic point is given by the mediant of the abscissae of the lower edge extremities, *i.e.*  $\underline{Q}$  and  $\bar{O}$  (direct consequence of Property 1): on line 5, the mediant is computed, and the point of  $R$  with this abscissa is the characteristic point.

---

**Algorithm 2.** Find the characteristic point

---

Let  $R(x, y)$  be the ray output by Algorithm 1;

Let  $M(\frac{p}{q}, \frac{r}{q})$  and  $N(\frac{p'}{q'}, \frac{r'}{q'})$  belonging to  $R$ ;

- 1 Let  $\frac{p}{q}$  and  $\frac{\bar{p}}{q}$  be the fractions before  $\frac{p}{q}$  and after  $\frac{p'}{q'}$  in the Farey Series of order  $\max(x, n - x)$ .
  - 2 Let  $\underline{Q}$  (resp.  $\bar{O}$ ) be the intersection point between  $\alpha = \frac{p}{q}$  (resp.  $\frac{\bar{p}}{q}$ ) and  $R$ ;
  - 3 **if**  $x - q < 0$  **then** Return  $\underline{Q}$  **else**
  - 4     **if**  $x + \bar{q} > n$  **then** Return  $\bar{O}$  **else**
  - 5     | Let  $\frac{\hat{p}}{q} = \frac{p+\bar{p}}{q+\bar{q}}$ . Return the intersection point between  $\alpha = \frac{\hat{p}}{q}$  and  $R$ ;
- 

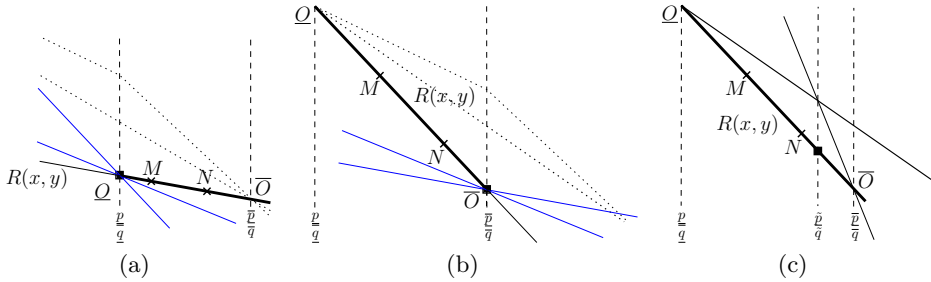
#### 4.4 Complexity

**Lemma 1.** *The complexity of the algorithm described in Section 4 is in  $\mathcal{O}(\log(n))$ , where  $n$  is the length of the DSS.*

*Proof.* We assume a computing model where standard arithmetic operations are done in constant time. Finding the ladder is done using the algorithm of Charrier and Buzer [3] that has a complexity of  $\mathcal{O}(\log(n))$ .

The localization of a lower edge is done with Algorithm 1: the computation of the  $R_i$  (line 2) is done in constant time thanks to Corollary 2, such that the global complexity of lines 2 and 3 is  $\mathcal{O}(\log(q))$  with  $q \leq n$ . The operations done in lines 4 to 7 are done in constant time, and the complexity of Algorithm 1 is in  $\mathcal{O}(\log(q))$ .

Algorithm 2 performs the last step of the algorithm. On line 1, two calls to the extended Euclidean algorithm are necessary to compute the lower edge extremities, which takes  $\mathcal{O}(\log(n))$ . All the other operations of this algorithm take  $\mathcal{O}(1)$ , which ends the proof.



**Fig. 4.** Three cases for the lower edge  $[Q\bar{O}]$ : (a) all the rays passing through  $Q$  (in blue) have a slope greater than  $x$  and  $Q$  is the characteristic point; (b) all the rays passing through  $\bar{O}$  (in blue) have a slope smaller than  $x$  and  $\bar{O}$  is the characteristic point; (c) neither  $Q$  nor  $\bar{O}$  is the solution, and the characteristic point is found with a median computation

Algorithm 2 can actually be optimized so that the call to the extended Euclidean algorithm (line 1) is not always necessary. These optimizations consist in the study of particular cases that are not presented here to keep the algorithm as clear as possible. All in all they do not change the theoretical complexity but lower the constant term, and slightly improve the practical efficiency.

This algorithm solves Problem 2 in  $\mathcal{O}(\log(n))$  where  $n$  is the order of the Farey fan. From the equivalence of Problems 1 and 2, this algorithm also solves Problem 1 in logarithmic time where  $n$  is the length of the DSS.

## 5 Experimentation

We have implemented the presented algorithm using the open-source library `DGtal` [1]<sup>1</sup>. The algorithm is very easy to implement and does not require continuous fractions implementation as in [14,15]. The algorithms of Said and Lachaud [14,15] being implemented in this library, comparing the algorithms was then an easy task. We also conducted the experimentation along the same protocol as the one they proposed as a test file in `DGtal`. Basically, the idea is to randomly choose a maximal value  $N$  for the parameter  $b$  of the DSL ( $a$  is smaller than  $b$ ), then fix a maximal value for the length  $n$  of the DSS, and finally randomly choose a shift  $\mu$  and the abscissa of the DSS first point. Each experiment is conducted for 10000 randomly chosen parameters.

The algorithms are executed to compute the characteristics of the DSS contained in the DSL. For each algorithm, the total running time is measured and divided by the total number of trials.

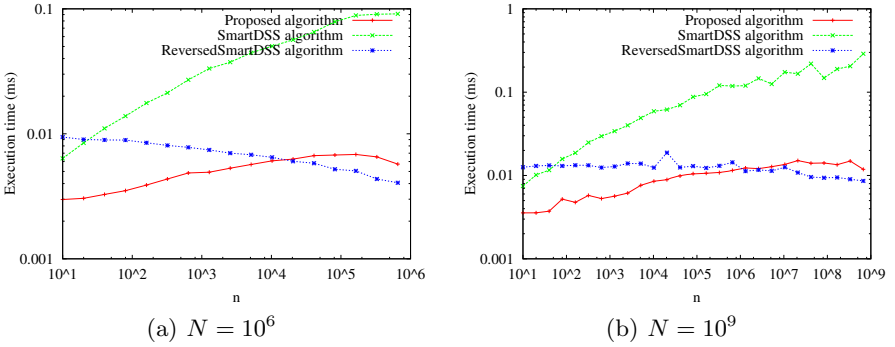
Figure 5 represents the results obtained for  $N = 10^6$  in (a),  $N = 10^9$  in (b) and  $n$  taking all the value of the form  $10.2^k$  in the interval  $[10, N]$ . The graph represents the execution time in ms versus the maximal length  $n$  of the DSS.

<sup>1</sup> The C++ code of this algorithm is freely available on the webpage <http://www.gipsa-lab.grenoble-inp.fr/~isabelle.sivignon>



The first observation is that SmartDSS is clearly slower than the other two algorithms. The second observation concerns the behaviour of the curves: the execution time increases with  $n$  for our algorithm while it decreases for ReversedSmartDSS. This is consistent with the complexities of the algorithms. The complexity of our algorithm is logarithmic in the length of the DSS while the complexity of ReversedSmartDSS depends on the difference of depth of the slope of the DSL and the slope of the DSS. Consequently, our algorithm is more efficient for short DSSs, while for ReversedSmartDSS, the greater is  $n$ , the smaller is the difference of slopes, and the more efficient is the algorithm.

It is thus interesting to study the value of  $n$  for which the two curves cross each other. We see in Figure 5 that this value is  $10^4$  for  $N = 10^6$  and  $10^6$  for  $N = 10^9$ . Other experiments show that this value is  $10^3$  for  $N = 10^4$  and that the threshold ratio  $\frac{n}{N}$  below which our algorithm is faster tends to decrease with  $N$ .



**Fig. 5.** Runtime comparison of our algorithm and the algorithms of [14,15]

## 6 Conclusion

We have proposed an algorithm to compute the characteristics of a DSS which is a subsegment of a DSL of known characteristics. We use the Farey fan and its numerous arithmetical properties to design a very efficient both theoretically and practically, and easy to implement algorithm to solve this problem.

We presented the algorithm in the case where the DSL parameters are rational fractions. However, it can be straightforwardly extended to deal with irrational parameters.

The experimental section has shown that our algorithm is faster than the ReversedSmartDSS algorithm when the length of the DSS is sufficiently smaller than the DSL period. This suggests that the ReversedSmartDSS algorithm should be preferred when the DSL parameters are issued from the recognition of a DSS on an image, and that our algorithm would perform better to draw a DSS given by floating-point vertices on an image. It would however be interesting to deepen this comparison.

## References

1. DGtal: Digital Geometry Tools and Algorithms Library, <http://libdgtal.org>
2. Berenstein, C., Lavine, D.: On the number of digital straight line segments. *IEEE Trans. on Pattern Anal. and Mach. Intell.* 10(6), 880–887 (1988)
3. Charrier, E., Buzer, L.: Approximating a real number by a rational number with a limited denominator: A geometric approach. *Discrete Applied Mathematics* 157(16), 3473–3484 (2009)
4. Coeurjolly, D., Sivignon, I., Dupont, F., Feschet, F., Chassery, J.-M.: On digital plane preimage structure. *Discrete Applied Mathematics* 151(1-3), 78–92 (2005)
5. Debled-Rennesson, I., Reveillès, J.-P.: A linear algorithm for segmentation of digital curves. *Inter. Jour. of Pattern Recog. and Art. Intell.* 9(6), 635–662 (1995)
6. Dorst, L., Smeulders, A.N.M.: Discrete representation of straight lines. *IEEE Trans. on Pattern Anal. and Mach. Intell.* 6(4), 450–463 (1984)
7. Dorst, L., Smeulders, A.W.: Discrete straight line segments: Parameters, primitives and properties. In: *Vision Geometry. Contemporary Mathematics*, pp. 45–62. Am. Math. Soc. (1991)
8. Hardy, G.H., Wright, E.M.: *An Introduction to the Theory of Numbers*. Oxford Society (1989)
9. Klette, R., Rosenfeld, A.: *Digital geometry - geometric methods for digital picture analysis*. Morgan Kaufmann (2004)
10. Koplowitz, J., Lindenbaum, M., Bruckstein, A.: The number of digital straight lines on an  $n \times n$  grid. *IEEE Trans. on Info. Theory* 36(1), 192–197 (1990)
11. Kovalevsky, V.: New definition and fast recognition of digital straight segments and arcs. In: *Inter. Conf. on Patt. Anal. and Mach. Intell.*, vol. 2, pp. 31–34 (1990)
12. McIlroy, M.D.: A Note on Discrete Representation of Lines. *AT&T Technical Journal* 64(2), 481–490 (1985)
13. Wein, R., Fogel, E., Zukerman, B., Halperin, D.: *CGAL - 2D Arrangements*, [http://www.cgal.org/Manual/3.3/doc\\_html/cgal\\_manual/Arrangement\\_2/Chapter\\_main.html](http://www.cgal.org/Manual/3.3/doc_html/cgal_manual/Arrangement_2/Chapter_main.html)
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)
15. Said, M., Lachaud, J.-O.: Computing the Characteristics of a SubSegment of a Digital Straight Line in Logarithmic Time. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) *DGCI 2011. LNCS*, vol. 6607, pp. 320–332. Springer, Heidelberg (2011)
16. Sivignon, I., Dupont, F., Chassery, J.M.: Digital intersections: minimal carrier, connectivity and periodicity properties. *Graphical Models* 66(4), 226–244 (2004)
17. Troesch, A.: Interprétation géométrique de l'algorithme d'euclide et reconnaissance de segments. *Theor. Comput. Sci.* 115(2), 291–319 (1993)

# Persistent Patterns in Integer Discrete Circles

André Hoarau and Thierry Monteil

CNRS – Université Montpellier 2

<http://www.lirmm.fr/~hoarau>,

<http://www.lirmm.fr/~monteil>

**Abstract.** We study patterns that appear in discrete circles with integer center and radius. As the radius goes to infinity, the patterns get closer to digital straight segments: the notion of tangent words (described in Monteil DGCI 2011) allows to grasp their shape. Unexpectedly, some tangent convex words do not appear infinitely often due to deep arithmetical reasons related to an underlying Pell-Fermat equation. The aim of this paper is to provide a complete characterization of the patterns that appear in integer discrete circles for infinitely many radii.

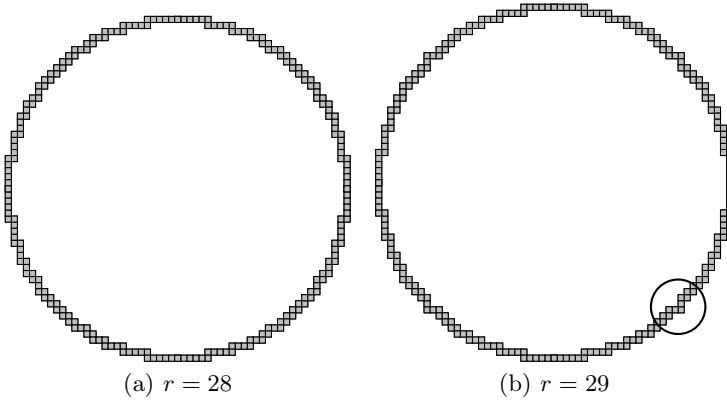
**Keywords:** discrete circle, asymptotics, digital straight segment, tangent word, Pell-Fermat equation.

## 1 Introduction

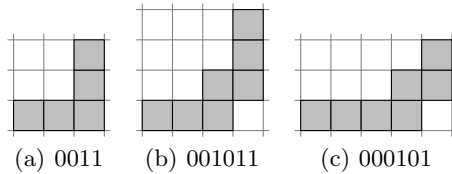
Freeman digitization schemes allow to associate a set of pixels to a planar object and a sequence of adjacent pixels to a planar curve in very natural ways ; the Freeman code associates a finite word to a sequence of adjacent pixels [6]. The case of straight segments is well understood, and the associated words are known to be the balanced words [8], [11]. We are interested here in the words appearing in the Freeman code of another classical geometrical object: the discrete circles [3], [1], [7], [5].

In 1979, Zenon Kulpa [7] noticed that some “spikes” appear on the diagonal for arbitrary big radii in the Grid Intersect Quantization digitization of integer circles. For big radii, such spikes look unnatural since, as the curvature of circles of big radii goes to zero, we expect the digitization of the big circle to look locally like digital straight segments. Unfortunately, Kulpa could not go further since his remark is based on a visual description about the angle between three consecutive pixels.

In this paper, we shall use another digitization scheme, namely the Square Box Quantization (SBQ) described in [6] (see also Section 2.1). As we shall see, the same phenomenon appears in this case (Fig. 1). It turns out that many finite patterns not corresponding to digital straight segments do appear in integer circles for arbitrary large radii. For example, the patterns of Fig. 2 could be considered as the “next spikes”, (b) is obtained by stretching (a), (c) is obtained by shearing (a).



**Fig. 1.** The spike coded by 0011 appears in the digitization of big integer circles



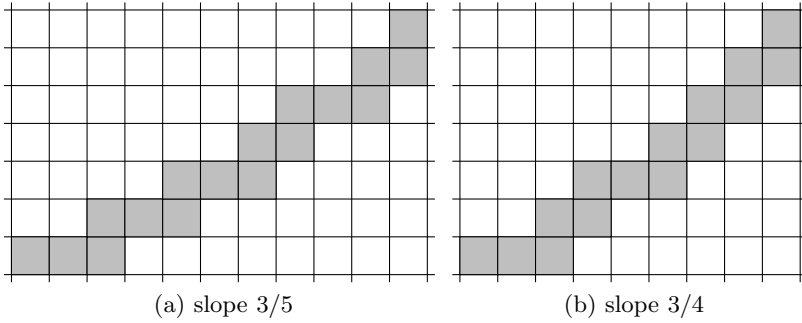
**Fig. 2.** The most elementary spikes

A word is said to be *persistent* if it appears in the Freeman code of integer discrete circles for infinitely many radii. Their complete description is the aim of this paper.

As we can consider the dual point of view where the circle radius does not grow but the grid mesh vanishes, we know that persistent words are tangent convex words, which were introduced in [9]. Experimentations lead us to the observation of an unexpected behaviour: while most tangent convex words seem to be persistent, some of them are not (Fig. 3).

As we shall see, the difference between both examples of Fig. 3 relies on a deep arithmetical reason. We can define a rational slope  $p/q$  for non-balanced tangent convex words, and say that it is *Pythagorean* if  $p^2 + q^2$  is a square. Knowing whether a tangent convex word is persistent or not is related to a system of Pell-Fermat inequalities whose main parameter depends on the slope of the word. For example, the word (b) is not persistent since its slope  $3/4$  is Pythagorean ( $3^2 + 4^2 = 5^2$ ). We will prove the following

**Theorem 1.** *A word is persistent if, and only if, it is tangent convex with a non-Pythagorean slope. In particular, balanced words are persistent.*



**Fig. 3.** Two tangent convex words: (a) is persistent, (b) is not

## 2 Preliminaries and Tools

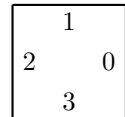
### 2.1 Framework

Discrete geometry introduces various schemes to associate a set of pixels to a plane continuous object. When the object is a curve, the pixels are ordered and the Freeman code associates to such a sequence a word over the alphabet  $\mathbb{Z}/4\mathbb{Z}$  or  $\mathbb{Z}/8\mathbb{Z}$  depending on whether two consecutive pixels have to share an edge or only a vertex. For example, the Freeman codes of digital straight segments are known to be precisely the *balanced* words using two consecutive letters (see e.g. [8], [11]), no matter the code or the digitization scheme. The case of discrete circles is wilder, there are various ways to define a discrete circle: the set of pixels can be described in an algorithmic way like in [3] or [13], it can also be described as the set of solutions of some analytic equation like in [1] or [5].

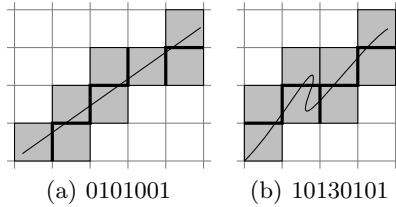
The model we are considering is the *Square Box Quantization* (SBQ) of integer circles [6]. The SBQ of a curve is the set of pixels that it intersects. An *integer circle* is a circle with integer radius  $r$  and centered at  $(0, 0)$ . It is denoted by  $C(r)$ .

Note that an integer circle cannot meet any pixel vertex since such points have coordinates of the form  $(p + 1/2, q + 1/2)$  for some  $(p, q) \in \mathbb{Z}^2$  and  $(p + 1/2)^2 + (q + 1/2)^2$  cannot be an integer. In such a non-ambiguous case, the Square Box Quantization is also known as the *supercover* [4] or the *standard model* [2].

The Freeman code associated to the Square Box Quantization of a curve  $\gamma$  also corresponds to what dynamicists call the *cutting sequence*  $c(\gamma)$ : it is obtained by reading the letters associated to the edges (see figure on the right) of the pixels that  $\gamma$  intersects along the time. Such a word defines a *pattern*, that is, a finite set of pixels defined up to an integer translation.



Hence, to see if a word  $u$  belongs to the cutting sequence of an integer circle of radius  $r$ , we have to ensure that, for some integer translation, each edge of the associated translated pattern corresponding to each letter is crossed by the



**Fig. 4.** Examples of cutting sequences, and their associated pattern (in gray)

circle. This is done by checking that, for each such edge, the distance from the center to one of the vertices of the edge is less than  $r$ , and the distance from the center to the other vertex is greater than  $r$ . Those vertices play a central role and are called *control points* (see Fig. 9).

Since the center of an integer circle is also the center of a pixel, the global symmetries allow us to restrict our study to the seventh octant, where the Freeman code only uses the letters 0 and 1, and the slope is between 0 and 1.

## 2.2 Tangent Convex Words

Tangent convex words were described in [9] Section 4.5. We recall their structure in a self-contained way and explain the geometric conditions that ensure their existence as a persistent word.

Let  $\mathcal{C}^+$  denote the set of *convex curves*, that is, the set of smooth regular curves with positive curvature. When  $\gamma$  belongs to  $\mathcal{C}^+$ , we denote by  $T(\gamma)$  the set of words that appear in the coding of  $\gamma$  with grids of arbitrary small meshes:

$$T(\gamma) = \bigcap_{\epsilon > 0} \bigcup_{\text{mesh}(G) \leq \epsilon} F(\gamma, G) \quad (1)$$

where  $F(\gamma, G)$  denotes the set of factors appearing in the Freeman code of  $\gamma$  by the grid  $G$ . A word  $u$  is *tangent* to a curve  $\gamma$  if  $u \in T(\gamma)$ . The *tangent convex words* are all the words which are tangent to some convex curve:  $T(\mathcal{C}^+) = \bigcup_{\gamma \in \mathcal{C}^+} T(\gamma)$ . A non-balanced tangent convex word is called a *spike*.

The relationship with integer circles is the following: up to renormalization, we can consider that we code a single unit circle centered at  $(0, 0)$  with grids of meshes  $1/n$  ( $n \in \mathbb{N}^*$ ). Since the unit circle (parametrized counterclockwise) belongs to  $\mathcal{C}^+$ , we deduce that the persistent words of integer circles form a subset of the set of tangent convex words  $T(\mathcal{C}^+)$ . This allow us to restrict our search among this set, which is well structured and has small complexity: the number of elements of  $T(\mathcal{C}^+)$  of length  $n$  is equivalent to  $n^3/6$  [10].

Let  $\gamma = (\gamma_x, \gamma_y) : [0, 1] \rightarrow \mathbb{R}^2$  be a convex curve. With almost no loss of generality and since it corresponds to the seventh octant assumption done in Subsection 2.1, we assume that  $\gamma$  is going in the North and East direction, that



**Fig. 5.**  $u = 00001000100100101001$

is,  $\forall t \in [0, 1], \gamma'_x(t) > 0$  and  $\gamma'_y(t) > 0$ . The coding of such curves only uses the letters 0 and 1 (Fig. 5). For any  $t \in [0, 1]$ , we denote the slope of  $\gamma$  at  $t$  by  $\rho(\gamma'(t)) = \frac{\gamma'_y(t)}{\gamma'_x(t)}$ .

Let  $u$  be a tangent word of  $\gamma$ : there exists a sequence  $(G_n)$  of grids whose meshes go to 0 and such that  $\forall n \in \mathbb{N}, u \in F(\gamma, G_n)$ . In particular, for any integer  $n$ , there exist two sequences  $(t_n^1), (t_n^2)$  in  $[0, 1]$  such that  $u$  is the Freeman code of  $\gamma|_{]t_n^1, t_n^2[}$  with respect to the grid  $G_n$ . Up to taking a subsequence (the segment  $[0, 1]$  is compact), we can assume that  $(t_n^1)$  and  $(t_n^2)$  both converge to some  $t \in [0, 1]$ : we say that  $\rho(\gamma'(t))$  is a *slope* of  $u$  (it may not be unique since  $u$  can be tangent to various curves).

**Slope Different from 1: Desubstitution.** Assume that  $\rho(\gamma'(t)) < 1$ . This implies that the word 11 is not a factor of  $u$ . We can use this information to construct a word  $\delta_0(u)$  from  $u$  such that  $|\delta_0(u)| < |u|$  and  $\delta_0(u) \in T(\delta_0(\gamma))$ , where  $\delta_0(\gamma)$  is another convex curve and  $T$  is defined by equation 1. For each  $n$ , we add diagonal edges to the grid  $G_n$  which we label by the letter 5. Hence, we can associate a new cutting sequence  $u'$  by inserting the letter 5 between any two consecutive 0 (Fig. 6).

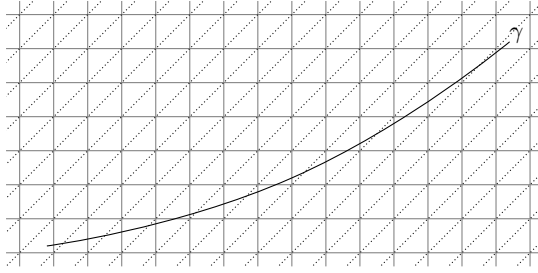
Then, we remove the vertical lines of the grid, and associate the corresponding word  $u''$  obtained from  $u'$  by removing the letter 0 (Fig. 7).

We then renormalize the parallelogram grid back to  $G_n$  by applying the shear matrix

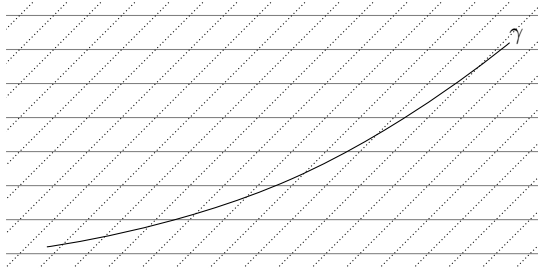
$$M_0 = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}^{-1}.$$

The word obtained by replacing the letter 5 by 0 in  $u''$  is denoted by  $\delta_0(u)$  and is a factor of  $F(\delta_0(\gamma), G_n)$ , where  $\delta_0(\gamma) = M_0 \circ \gamma$  is the curve obtained by applying the matrix  $M_0$  to  $\gamma$  (Fig. 8).

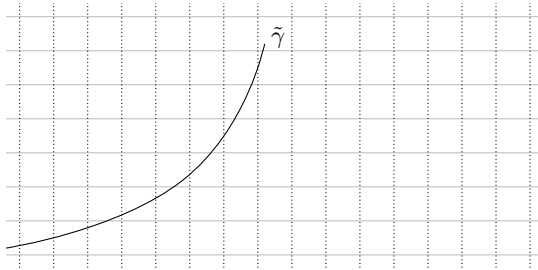
The matrix  $M_0$  corresponds to a linear bijective bi-uniformly continuous map, it therefore preserves the regularity of the curve. Moreover, since it has positive determinant, the sign of the curvature of the curve is preserved:  $\delta_0(\gamma)$  belongs to  $\mathcal{C}^+$ . All the operations are reversible, hence  $u$  is in  $T(\gamma)$  if, and only if,  $\delta_0(u)$  is in  $T(\delta_0(\gamma))$ .



**Fig. 6.**  $u' = 050505010505010501050105010501$



**Fig. 7.**  $u'' = 55515515151151$



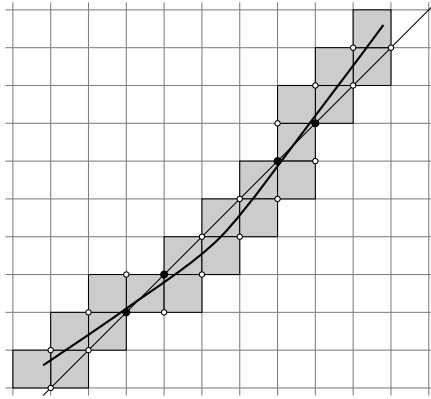
**Fig. 8.**  $\delta_0(u) = 00010010101101$

In the same way, if  $\rho(\gamma'(t)) > 1$  then 00 is not a factor of  $u$ , we insert a 5 between two consecutive 1, and renormalize to obtain a word  $\delta_1(u)$ , and the matrix used for the renormalization is

$$M_1 = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}^{-1}.$$

**Slope Equal to 1: Diagonal Words.** If  $\rho(\gamma'(t)) = 1$ , we say that the word  $u$  is *diagonal*. For large  $n$ , the curve  $\gamma$  cannot intersect more than one integer diagonal (a line of slope 1 passing through a corner of  $G_n$ ). As explained in





**Fig. 9.** A diagonal word and its principal (black) and secondary (white) control points

[9], the diagonal word can oscillate a lot for smooth curves, but in the case of a convex curve, the oscillation is very limited: for  $n$  large enough, the curve  $\gamma$  intersects the integer diagonal at most twice between  $t_n^1$  and  $t_n^2$  (Fig. 9).

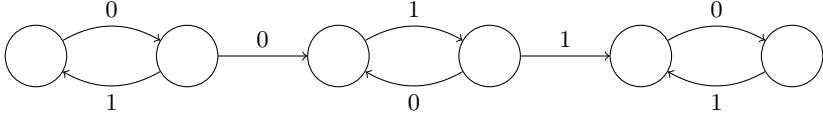
When the number of such intersections is less than 2, we get a balanced word. When this number is 2, the first intersection corresponds to an occurrence of 00 in  $u$ , and the second corresponds to an occurrence of 11 (in particular the slope of  $u$  is unique and must be equal to 1). The remaining of the diagonal word  $u$  of  $T(\mathcal{C}^+)$  is made by an alternation of 0 and 1, hence it is completely determined by the three lengths: before 00, between 00 and 11, after 11.

The grid points on the integer diagonal surrounding the two intersections are called *principal control points* because they contain all the information concerning the localization of the convex curve coded by  $u$  for  $n$  large enough. The other control points are said to be *secondary* (the satisfaction of those control points — that is, the satisfaction of the equations derived from those control points — is guaranteed by the fact that the curve is convex or by the fact that, when  $n$  is large enough, the curve becomes very close to the integer diagonal).

The minimal non-balanced diagonal words of  $T(\mathcal{C}^+)$  are of the form  $u = 00(10)^\ell 11$  for some integer  $\ell$  which is called the *width* of the spike  $u$ . Any non-balanced diagonal word of  $T(\mathcal{C}^+)$  can be extended in a single way to the left and to the right, by adding alternately a 0 or a 1. Diagonal words of  $T(\mathcal{C}^+)$  are the words recognized by the non-deterministic automaton depicted in Fig. 10, where all states are considered as initial and final.

The two transitions without return correspond to the two possible intersections with the integer diagonal, i.e. to the occurrences of 00 and 11. Both transitions are used in recognizing  $u$  if, and only if,  $u$  is not balanced. The width  $\ell$  corresponds to the number of central loops made during the recognition of  $u$ .

**Iteration.** Those considerations allow us to recognize the words of  $T(\mathcal{C}^+)$ : given a word  $u$ , we apply  $\delta_0$  (resp.  $\delta_1$ ) to  $u$  as long as 11 (resp. 00) does not appear in  $u$ . The length of the considered words is strictly decreasing, hence after finitely many steps, we reach one of the following two cases:



**Fig. 10.** Automaton recognizing diagonal words of  $T(\mathcal{C}^+)$

1. the word is empty. Hence, the initial word  $u$  is balanced [8], therefore it belongs to  $T(\mathcal{C}^+)$ .
2. 00 and 11 both appear in the word: it suffices to check that it is recognized by the automaton of Fig. 10 to conclude that the initial word  $u$  is in  $T(\mathcal{C}^+)$ .

This provides a desubstitution algorithm which recognizes the words of  $T(\mathcal{C}^+)$ .

Note that when a word  $u \in T(\mathcal{C}^+)$  is not balanced, then the iterated desubstitution leads to a diagonal word where both 00 and 11 appear: *the slope* of  $u$  is then unique, and independent of any supporting curve  $\gamma$ , time  $t$  or sequence of grids  $(G_n)$ . It can be computed using the continued fraction associated to the alternation of  $\delta_0$  and  $\delta_1$  leading to the diagonal word whose slope equals 1. For balanced words however, the set of possible slopes is an interval of nonempty interior.

Conversely, we can generate the words of  $T(\mathcal{C}^+)$  by going backward in the algorithm: we construct a diagonal word by describing a path in the automaton of Fig. 10. Then we apply one of the following operations, as much as we want:

1. add a 0 to the right of each 1,
2. add a 0 to the left of each 1,
3. add a 1 to the right of each 0,
4. add a 1 to the left of each 0.

The words of  $T(\mathcal{C}^+)$  are the factors of the words we obtain that way. It is important to notice that the primary control points can be followed during this construction by applying to the initial primary control points the matrix  $M_0^{-1}$  in the first two cases and the matrix  $M_1^{-1}$  in the last two cases. Hence, we know how to localize a curve around a word of  $T(\mathcal{C}^+)$  of any slope.

### 2.3 Generalized Pell-Fermat Equations

A generalized Pell-Fermat equation is a Diophantine equation of the form  $x^2 - Dy^2 = K$ , where  $D$  is a positive integer parameter,  $K$  is an integer parameter, and  $x$  and  $y$  are the integer unknowns. When  $D$  is not a square, and when the equation admits at least one solution, then it admits infinitely many solutions [12]. The set of solutions is a finite union of geometric progressions.

## 3 Main Result

**Theorem 1.** *A word is persistent if, and only if, it is tangent convex with a non-Pythagorean slope. In particular, balanced words are persistent.*

We shall first deal with the spikes, which are somehow more rigid (they have a single slope and can be studied through a few principal control points), and then we will extend the obtained results to balanced words.

### 3.1 Spikes Crossing the Octants: Slopes 0 and 1

Due to the lack of space, the description of persistent spikes for the extremal slopes 0 (no spike of the form  $0^k 30^\ell 10^m$  is persistent) and 1 (all spikes of the form  $(01)^k 00(10)^\ell 11(01)^m$  are persistent) is postponed, we will concentrate on the general case of slopes in the open interval  $(0, 1)$  which contains all the ideas. We can still notice that the behaviour of those two particular cases is coherent with this general description, since  $0/1$  is Pythagorean and  $1/1$  is not.

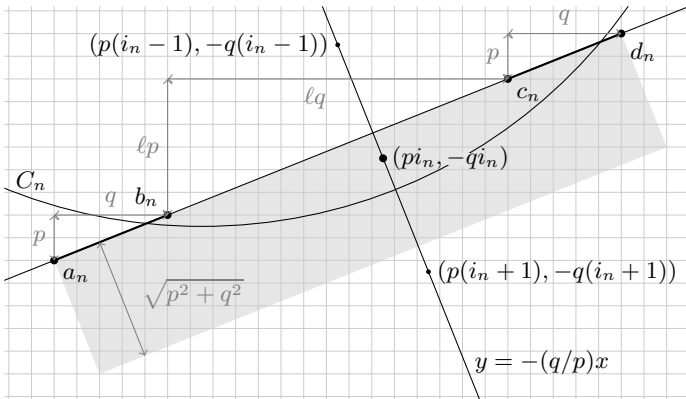
### 3.2 Spikes with Pythagorean Slopes in $(0, 1)$

**Lemma 1.** *A spike with a Pythagorean slope in  $(0, 1)$  is not persistent.*

*Proof.* Let  $u$  be a spike of slope  $p/q$ , such that  $0 < p/q < 1$  and  $p^2 + q^2 = s^2$  for some integer  $s$ . We want to prove that  $u$  is not persistent. Assume by contradiction that for an increasing sequence  $(r_n)$  of positive integers,  $u$  appears in the cutting sequence of the circle  $C_n = C(r_n)$ .

The position of the principal control points  $a_n, b_n, c_n, d_n$  of  $u$  relative to  $C_n$  is depicted in Fig. 11 (it is obtained from Fig. 9 by applying shear matrices).

Since  $C_n$  crosses both segments  $[a_n, b_n]$  and  $[c_n, d_n]$ , the line of equation  $y = -(q/p)x$  crosses the segment  $[a_n, d_n]$ , so we have a rough localization of the pattern associated to  $u$ . On this line, the points of the form  $(pi, -qi)$  where  $i$  is an integer are at distance  $\sqrt{p^2 + q^2}$  from each other. Hence, for each  $n$ , there is a point of coordinate  $(pi_n, -qi_n)$  in the rectangle whose base is  $[a_n, d_n]$  and whose height has length  $\sqrt{p^2 + q^2}$  (depicted in gray on the figure).



**Fig. 11.** Situation around the pattern associated to  $u$  for  $C_n$

Let  $(x_n, y_n)$  be the vector from  $(pi_n, -qi_n)$  to the point  $a_n$ . Since  $(pi_n, -qi_n)$  has integer coordinates and stays in a bounded rectangle around the pattern associated to  $u$ , the vector  $(x_n, y_n)$  takes only finitely many values. Up to taking a subsequence, we can assume that it is constant. Since  $(pi_n, -qi_n)$  is the center of a pixel and  $a_n$  is a corner of a pixel, there exists integers  $X$  and  $Y$  such that, for any  $n$ ,  $(x_n, y_n) = (X + 1/2, Y + 1/2)$ .

Since the point  $a_n$  is outside the circle  $C_n$ , we have

$$r_n^2 \leq (pi_n + X + 1/2)^2 + (-qi_n + Y + 1/2)^2$$

which can be rewritten as  $R_n^2 - I_n^2 \leq K_a$ , where  $R_n = 2sr_n$ ,  $I_n = 2s^2i_n + (2X + 1)p - (2Y + 1)q$  and  $K_a = ((2X + 1)p - (2Y + 1)q)^2 + (X + 1/2)^2 + (Y + 1/2)^2$ .

The same argument with the point  $b_n$  inside the circle  $C_n$  leads to an inequality of the form  $K_b \leq R_n^2 - I_n^2$ , where  $K_b$  is another constant ( $R_n$  and  $I_n$  are the same thanks to a nice cancellation).

Now, both sequences  $(R_n)$  and  $(I_n)$  are nonnegative integer sequences tending to infinity. Since the difference between their squares is bounded, it eventually equals zero. Hence, for  $n$  large enough,  $R_n = I_n$ . Since  $(p, q, s)$  is a primitive Pythagorean triple, exactly one of  $p$  or  $q$  is odd, hence  $I_n$  is odd. Unfortunately,  $R_n$  is even, a contradiction.

### 3.3 Spikes with non-Pythagorean Slopes in $(0, 1)$

**Lemma 2.** *A spike with a non-Pythagorean slope in  $(0, 1)$  is persistent.*

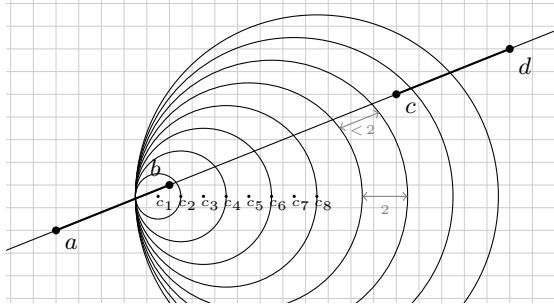
*Proof.* Let  $u$  be a spike of slope  $p/q$ , such that  $0 < p/q < 1$  and  $p^2 + q^2$  is not a square. Let  $\ell$  be its width. We want to prove that  $u$  is persistent.

We start by finding an integer circle  $C$  which is well located with respect to the principal control points of  $u$ , that is, we require that  $C$  crosses both segments  $[a, b]$  and  $[c, d]$  (we do not care about the secondary control points, hence  $u$  need not be a factor of the cutting sequence of  $C$  but this will become true for larger circles).

Let  $c_1$  denote the center of the pixel located down-left of the control point  $b$  (see Fig. 12), and let  $C_1$  denote the circle centered at  $c_1$  with radius 1. For any positive integer  $i$ , let  $c_i$  denote the integer point  $c_1 + (i - 1, 0)$ , and let  $C_i$  denote the circle with center  $c_i$  and radius  $i$ .

By construction, each  $C_i$  is an integer circle that crosses the segment  $[a, b]$ . There is another intersection between  $C_i$  and the line defined by  $(a, d)$ : from  $i$  to  $i + 1$ , this intersection goes forward in the direction from  $a$  to  $d$  and the distance between two intersections is a positive constant strictly less than 2. Since the length of the segment  $[c, d]$  is  $\sqrt{p^2 + q^2} > \sqrt{5} > 2$ , there is an integer  $i$  such that  $C_i$  crosses  $[c, d]$ . Let  $c = c_i$  and  $C = C_i$ . The integer circle  $C$  satisfies the control points of  $u$ .

Now, for any integer radius  $r$ , we try to find a solution by locating the pattern associated to  $u$  in such a way that the point  $c$  coincides with some  $(pi, -qi)$  where  $i$  is an integer. Similarly to the case of Pythagorean slope, the fact that the circle



**Fig. 12.** Construction of an integer circle satisfying the principal control points of  $u$

of radius  $r$  is well located with respect to the control points is equivalent to the existence of some integer  $i$  such that

$$\max(K_b, K_c) \leq 4(p^2 + q^2)r^2 - I^2 \leq \min(K_a, K_d)$$

where  $I = 2(p^2 + q^2)i + (2X + 1)p - (2Y + 1)q$  and  $K_a, K_b, K_c, K_d$  are some constants.

Hence, the radii for which the circles are well located with respect to the principal control points of  $u$  and such that the point  $c$  corresponds to some  $(pi, -qi)$  satisfy a Pell-Fermat inequation, whose set of solutions is the union of the solutions of the family of generalized Pell-Fermat equations:  $4(p^2 + q^2)r^2 - I^2 = K$  for  $K$  in the set  $[\max(K_b, K_c), \min(K_a, K_d)] \cap \mathbb{Z}$ .

By construction of  $C$ , the pair  $(r_0, 0)$  is a particular solution of one of those equations. Hence, since  $D = 4(p^2 + q^2)$  is not a square, Subsection 2.3 ensures that that equation has infinitely many solutions: there are infinitely many radii  $r$  such that the circle  $C(r)$  is well located with respect to the principal control points of  $u$ . When such an  $r$  is large enough, the secondary control points become automatically satisfied as well, hence  $u$  appears in the cutting sequence of  $C(r)$ .

Therefore,  $u$  is persistent.

### 3.4 Balanced Persistent Words

**Lemma 3.** *Every balanced word is persistent.*

*Proof.* Let  $u$  be a balanced word. As already noticed, the set of slopes associated to  $u$  contains a open interval  $I$ , hence, by density, there exists a non-Pythagorean rational number  $p/q$  in  $I$  (it suffice to ensure that  $p$  and  $q$  are odd). Then, the word  $u$  can be extended to a spike of slope  $p/q$ , which is persistent. Hence,  $u$  is persistent, since it is the factor of a persistent word.

*Proof of Theorem 1.* Since persistent words are tangent convex, since the tangent convex words are either spikes or balanced, we just proved the Theorem 1.

*Decision algorithm.* As explained in [9], deciding whether a word belongs to  $T(\mathcal{C}^+)$  can be done in linear time. The computation of the slope can be done along the process described in Section 2.2, “Iteration”, hence deciding whether a given word is persistent can be done in linear time.

## References

1. Andres, E.: Discrete circles, rings and spheres. *Computers & Graphics* 18(5), 695–706 (1994)
2. Andres, E.: Defining Discrete Objects for Polygonalization: The Standard Model. In: Braquelaire, A., Lachaud, J.-O., Vialard, A. (eds.) *DGCI 2002*. LNCS, vol. 2301, pp. 313–325. Springer, Heidelberg (2002)
3. Bresenham, J.: A linear algorithm for incremental digital display of circular arcs. *Commun. ACM* 20(2), 100–106 (1977)
4. Cohen-Or, D., Kaufman, A.: Fundamentals of surface voxelization. *Graphical Models and Image Processing* 57(6), 453–461 (1995)
5. Fiorio, C., Toutant, J.-L., Jamet, D.: Discrete Circles: an arithmetical approach with non-constant thickness. In: *Vision Geometry XIV, IS&T/SPIE 18th Annual Symposium Electronic Imaging*, pp. 60660C01–60660C12. SPIE (January 2006)
6. Freeman, H.: Computer processing of line-drawing images. *Computing Surveys* 6, 57–97 (1974)
7. Kulpa, Z.: On the properties of discrete circles, rings, and disks. *Computer Graphics and Image Processing* 10(4), 348–365 (1979)
8. Lothaire, M.: Algebraic combinatorics on words. *Encyclopedia of Mathematics and its Applications*, vol. 90. Cambridge University Press, Cambridge (2002)
9. Monteil, T.: Another Definition for Digital Tangents. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) *DGCI 2011*. LNCS, vol. 6607, pp. 95–103. Springer, Heidelberg (2011)
10. Monteil, T.: The complexity of tangent words. In: Ambroz, P., Holub, S., Masáková, Z. (eds.) *WORDS*. EPTCS, vol. 63, pp. 152–157 (2011)
11. Pytheas Fogg, N.: Substitutions in dynamics, arithmetics and combinatorics. *Lecture Notes in Mathematics*, vol. 1794. Springer, Berlin (2002)
12. Robertson, J.P.: Solving the generalized Pell equation  $x^2 - Dy^2 = N$  (2004), Preprint available at <http://www.jpr2718.org/pell.pdf>
13. Tougne, L.: Circle Digitization and Cellular Automata. In: Miguet, S., Ubéda, S., Montanvert, A. (eds.) *DGCI 1996*. LNCS, vol. 1176, pp. 283–294. Springer, Heidelberg (1996)

# Comparison of Point Clouds Acquired by 3D Scanner

Natalia Dyshkant\*

Lomonosov Moscow State University,  
Faculty of Computational Mathematics and Cybernetics,  
GSP-1, 1-52, Leninskiye Gory, MSU, Moscow, 119991, Russian Federation  
natalia.dyshkant@gmail.com

**Abstract.** 3D representation of real objects surfaces can be used in applications of computer graphics, medicine, geoinformatics, etc. We consider a problem of measure introducing for comparing of point clouds acquired by different scanning acts and types of scanners and designing of computationally efficient algorithms for their computing. The solution supposes estimation of disparity measure for the fixed position and search of such position that the measure is minimal by solving optimization problem of surface matching. The algorithm for efficient localization of mesh nodes in a Delaunay triangulation is proposed. As the applications several problems of 3D face model analysis were considered.

**Keywords:** Discrete surface model, Delaunay triangulation, Euclidean minimum spanning tree, computational geometry, 3D face image.

## 1 Introduction

Information obtained from a 3D scanner after scanning of an objects's surface is usually presented as a discrete set of nodes with three-dimensional coordinates and represents a *discrete model* of a surface. The concepts of connectivity, topology, or continuous surface are not specified explicitly for such set. On basis of this set one can receive triangulated polygonal surface model [1], which will be a *continuous model* of a surface.

A literature review of existing techniques for single-valued surface modelling showed that there are two basic approaches to represent such surfaces: definition at the nodes of the regular or irregular meshes. A two-dimensional (plane) *mesh* is a set of mutually connected geometric elements (nodes, edges, and cells). The mesh nodes represent a finite set of points on the plane. Both approaches have their advantages and disadvantages. Method using an irregular mesh allows to adapt to the required accuracy of the surface description and does not contribute redundancy, which increases computing resources, to the source data.

Often during comparison of two surfaces it is assumed that for each node of one surface point there is the corresponding node of the second surface. This

---

\* The reported study was partially supported by RFBR, research projects Nos. 12-07-13307 mol\_a, 11-01-00783 a.

assumption implies definition of surfaces at the same discrete set of nodes. Initial surface objects acquired by three-dimensional scanning, have irregular structure. During the transforming stage from the original irregular data to regular mesh one face the challenge of choosing the optimal size for size of mesh cell, which leads to inefficiency of the approach.

The majority of the existing measures for comparing surfaces can be calculated directly if both surfaces are defined at the nodes of some common mesh. However, such measures do not allow extension to the case of surfaces defined on different meshes without the stage of transition to regular meshes. There are measures that can easily be generalized to the case of two different meshes, but their calculation in this case has a quadratic complexity.

The existing methods for mapping the surfaces can be divided into two classes:

- a. Surface adjustment based on calculation of distances between points in the three-dimensional space, which has a large computer complexity.
- b. Recalculation of the original data in a regular two-dimensional mesh, which leads to description redundancy and also to significant increase in computational complexity.

It follows that the problem of developing of new computationally efficient algorithms for comparison of surfaces represented as point clouds, preserving the original irregular meshes remains actual at present.

The paper is organized as follows. A literature review of the problem for comparison of point clouds is given in section 2. The proposed approach for surface comparison is given in section 3. The proposed method of mesh localization in Delaunay triangulation presented in Section 4. Section 5 presents some applied problems for 3D face image analysis considered by the author. Conclusion is given in section 6.

## 2 Related Work

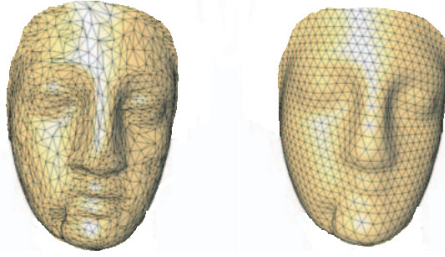
As it was stated above, many existing approaches for point cloud comparison use the transformation of initial irregular meshes to the common regular mesh [2], [3], [4] (see Fig. 1). After such transformation the approaches of surface comparison and matching for regular sets of nodes [5] can be applied.

During the transformation we face the problem of choosing the optimal step for a regular mesh, which leads to a significant amount of calculations to achieve an acceptable accuracy of the approximation surface and inefficiency. In this paper we propose a method for comparing surfaces with preservation of the original irregularity of meshes.

The most of existing methods is assumed that for each single node of the first surface there is the corresponding node on the second surface [6], [7]. There are methods based on a comparison of feature descriptions [8].

One of the basic algorithms for surface matching is the algorithm of iterative closest points — ICP, proposed in [6], [9], [7]. The algorithm uses an iterative procedure to minimize the average distance between two point clouds. This requires an initial rough estimate of converting one cloud to another, which is





**Fig. 1.** Transformation of the irregular mesh to the regular one (adoption of illustration from [4])

gradually refined in the process of minimization. For two given point clouds  $S_1$  and  $S_2$  the algorithm finds the transformation from  $S_1$  into  $S_2$ .

The ICP algorithm can be used to align the images of the same object obtained from different angles, which has common areas — regions of overlap. This assumes that there are pairs of closest points such that distances between them are less than a threshold in the regions of overlap. If some "wrong" pairs which do not belong to the region of overlap are included in a list of pairs of closest point, it will negatively affect to the results of the algorithm.

Let  $\{(s_i^1, s_i^2)\}_{i=1}^N$  be a set of pairs of closest points for  $S_1$  and  $S_2$ . During the algorithm procedure the average distance between the point clouds  $S_1$  and  $S_2$  is minimized:

$$E = \frac{1}{N} \sum_{i=1}^N d(s_i^1, s_i^2) \rightarrow \min, \quad (1)$$

where  $d(\cdot, \cdot)$  is the Euclidean distance between two points.

To calculate the distance between the point  $s^1 \in S_1$  and the cloud of points  $S_2$  in the original version of the algorithm [6] the "point-to-point" distance was used:

$$\rho(s^1, S_2) = \min_{s^2 \in S_2} d(s^2, s^1). \quad (2)$$

Metric (2) uses discrete surface models.

In [9] "point-to-plane" distance was proposed, the use of which implies that at each point of  $S_1$  and  $S_2$  normal to the surfaces specified by point clouds  $S_1$  and  $S_2$ , respectively, is known. Such information may be given initially, or calculated by averaging the normals of the adjacent triangles using triangulates polygonal approximation of the surface. In this case it the sum of squared distances between  $s_i^1$  and plane  $P_i^2$ , perpendicular to  $S_2$  at  $s_i^2$  over all pairs of nearest points  $(s_i^1, s_i^2)$  will be minimized:

$$E = \sum_{i=1}^N H^2(s_i^1, P_i^2) \rightarrow \min, \quad (3)$$

where  $H(s, P)$  — Euclidean distance between  $s$  and plane  $P$ .

General iterative scheme of the algorithm is the following:

1. Search of pairs of points  $(s_i^1, s_i^2)$ ,  $i = \overline{1, N}$  for the current mutual position of  $S_1$  and  $S_2$ .
2. Search of the transformation (parameters of translations and rotations) of point cloud  $S_1$ , reduces error  $E$  (1) and (3) using the least squares method. If the change of the error  $E$  is below a certain threshold, the algorithm terminates.
3. Application of the transformation found at the previous step to convert point cloud  $S_1$ . Transition to Stage 1.

Stages 1 — 3 are repeated while the reduction of the error exceeds a certain threshold value. The result is the final position of point cloud  $S_1$ .

Influence of "wrong" pairs of closest points is smaller for "point-to-plane" distance. But the disadvantage of the "point-to-plane" distance is that it is highly dependent on the initial relative position of  $S_1$  and  $S_2$  [10].

The main advantage of the ICP algorithm is its simple implementation, the disadvantage is a strong dependence on the initial approximation of objects, the computational complexity associated with search of all pairs of closest points of  $\{(s_i^1, s_i^2)\}_{i=1}^N$ .

Let  $N_1, N_2$  be the numbers of points in initial point cloud  $S_1, S_2$ , respectively. Then using a simple implementation the estimation of computational complexity of such search is  $O(N_1 N_2)$ , i.e. is quadratic if  $N_1 \approx N_2$ . Using more complex data structures — for example, k-d trees [11] — the search can be carried out in a time  $O(N_1 \log N_2)$ . Thus, the total number of operations required to find pairs of closest points in  $m$  iterations is  $O(m N_1 \log N_2)$ .

Large number of papers is devoted to various improvements of the algorithm:

- Modification of methods for selecting the region of overlap and pairs of closest points (for example, by introducing limitations to the class of motions that take one point to another [12], using the theory of random quantities [13], k-d trees [11], [7], genetic algorithms [14]);

- Introducing weights for pairs of closest points [15];

- Modification of the equation for the distance between two points [16], [17];

- Modification of the equation for the error to be minimized [18] and minimization procedure [17].

The drawback of the such improvements is their complexity, tuning to a specific experimental data, which reduces the robustness of the algorithms.

In [19] the problem of comparison of the surfaces for one and the same object, defined on different sets of points was considered. The distance from a point of one surface to the other surface was calculated along the normal to the second surface splines. Such measure is interesting because it does not require a reduction of the source data to common mesh. But the complexity of this approach is quadratic.

Such measures as, e.g., the average distance between the heights that can be directly calculated by recalculation of two surfaces at the nodes of common mesh, do not allow extensions to the case of their definition at two different meshes without the stage of transformation to regular meshes.

In [20] a surface matching algorithm based on minimizing the similarity measure between them was proposed. The compared surface  $S_1$  and  $S_2$  are considered as objects of linear space, and the similarity measure  $\rho(S_1, S_2)$  is the norm  $\|S_1 - S_2\|$  in this space. Let surface  $X$  be represented by triangulated piecewise linear model of  $N$  triangles,  $c_i$  — centroid of  $i$ -th triangle,  $n_i$  — normal vector to  $i$ -th triangle, the length of which is equal to the area of the triangle. The norm of surface  $X$  is introduced as

$$\|X\| = \sum_{i=1}^N \sum_{j=1}^N (n_i, n_j) e^{-|c_i - c_j|^2 / \sigma^2}. \quad (4)$$

Let the initial surfaces  $S_1, S_2$  be represented by triangulated piecewise linear model of  $N_1, N_2$  triangles, respectively;  $c_i^1(c_i^2)$  — centroid of  $i$ -th triangle of surface  $S_1(S_2)$ ,  $n_i^1(n_i^2)$  — normal vector to  $i$ -th triangle of surface  $S_1(S_2)$ , whose length is equal to the area of the triangle. The similarity measure proposed in [20], is defined as follows:

$$\begin{aligned} \rho(S_1, S_2) = & \sum_{i=1}^{N_1} \sum_{j=1}^{N_1} (n_i^1, n_j^1) e^{-|c_i^1 - c_j^1|^2 / \sigma^2} + \sum_{i=1}^{N_2} \sum_{j=1}^{N_2} (n_i^2, n_j^2) e^{-|c_i^2 - c_j^2|^2 / \sigma^2} - \\ & - 2 \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} (n_i^1, n_j^2) e^{-|c_i^1 - c_j^2|^2 / \sigma^2} = \|S_1 - S_2\|. \end{aligned} \quad (5)$$

Further, the value of the measure (5) is minimized by quickest descent method. The disadvantage of this approach is the quadratic complexity of similarity measure calculating.

### 3 Approach to Comparison of Point Clouds

The general formulation of the problem is the following: there exist two surfaces determined by the height functions at two finite sets of points; it is necessary to calculate some measure of similarity (or difference) between them.

Let  $T_1, T_2$ , and  $T$  be the Delaunay triangulations [21] constructed on the sets of nodes of the meshes  $g_1$  and  $g_2$  and the common mesh  $g = g_1 \cup g_2$ , respectively. Then  $T$  is a *common*, or *united*, *Delaunay triangulation*. Let us denote by  $Conv(g)$  the convex hull of mesh  $g$ . It is assumed that the initial meshes consist of non-intersecting sets of nodes.

#### 3.1 Proposed Measures

A measure for comparing surfaces determined by single-valued functions at the nodes of different non-regular triangulation meshes was proposed in [22]. The sum of volumes of the difference between the functions over all triangles of  $T$  was calculated in order to determine the measure. In this case, the values of

the initial functions  $f_1$  and  $f_2$  represented the surfaces were interpolated at the points of the opposite mesh using the localization of the triangulations  $T_1$  and  $T_2$  in each other. As a result of interpolation, we obtain the continuous functions  $\hat{f}_1$  and  $\hat{f}_2$  determined on the set  $Conv(g)$  for which  $\hat{f}_1 = f_1$ ,  $\hat{f}_2 = f_2$  at the nodes of  $g_1$ ,  $g_2$ , respectively. Let  $\mu(x, y) \geq 0$  be the function determining the weight of the difference of the surfaces at the point  $(x, y)$ . It is assumed that  $\mu(x, y)$  is defined and finite at all points of  $Conv(g)$  and is equal to zero beyond  $Conv(g)$ .

Let us introduce the weighted volume of the difference between the surfaces in the triangular region,

$$V_\mu(A, B, C, f_1, f_2) = \iint_{\Delta ABC} |\hat{f}_1(x, y) - \hat{f}_2(x, y)| \mu(x, y) dx dy. \quad (6)$$

For  $\mu = 1$ , the volume  $V_\mu$  is the metric  $L_1$  for the interpolated initial functions  $f_1$  and  $f_2$  on  $\Delta ABC$ . Let us denote by  $S_{Conv(g)}$  the area of the convex hull of set  $g$ , which is equal to the sum of areas of all triangles of the triangulation of set  $g$ . The following measure was proposed for comparison of surfaces:

$$\rho_{V_\mu}(f_1, f_2) = \sum_{\Delta ABC \in T} V_\mu(A, B, C, f_1, f_2) / S_{Conv(g)}, \quad (7)$$

Summing in (7) takes place over all triangles in  $T$ .

The proposed measure can be adapted for each particular application by introducing the function  $\mu$ .

In [23] special measure  $\rho_{dV}$  for case when two initial surface models have different level of detalization was proposed.

### 3.2 Surface Comparison Algorithm

An approach proposed for surface comparison and calculating measured is as follows: Delaunay triangulations are constructed on both meshes, then each of the functions is interpolated with respect to the other mesh, which is followed by the construction of the common triangulation for the two meshes. Then at each point of the merged meshes the values of the two functions are known, and operations can be performed on individual faces (triangles) of the common triangulation analyzing the mutual arrangement of the spatial triangles defined by the functions. Therefore, the developed algorithm is based on the idea of supplementing the values of each function in the nodes of the other mesh by constructing the Delaunay triangulation and their localization in each other.

Let us present main stages of the algorithm  $A^\rho$  for calculating measure  $\rho$ :

1. Construction of Delaunay triangulations  $T_1$ ,  $T_2$ ;
2. Construction of minimal spanning trees of Delaunay triangulations;
3. Localization of each of meshes  $g_1$ ,  $g_2$  in the triangulation constructed on the other mesh;
4. Interpolation of each of two functions  $f_1$ ,  $f_2$  on the mesh that the other function defined on;

5. Construction of the common triangulation  $T$  (see the linear merging method proposed in [24]);
6. Comparison of the functions on the separate cells of triangulation  $T$  (calculating measure  $\rho$ ).

### 3.3 Computational Complexity

The numerical complexity of algorithm  $A^\rho$  is  $O(n \log n)$ , where  $n$  is the total number of points of mesh  $g$ . It was proved that in case of Delaunay triangulations  $T_1$  and  $T_2$  are constructed at the preprocessing stage, the *expected time* to compute measure  $\rho_{V_\mu}$  is  $O(n)$ . At the same time in the mentioned conditions the numerical complexity of calculating measure  $\rho_{dV}$  is  $O(n)$  *in the worst case* (see the proof in [23]). The key stage of the algorithm that influenced on running time is the stage of mesh localization. The problem of computational complexity of stage 3 will be considered in Section 4.

### 3.4 Surface Matching

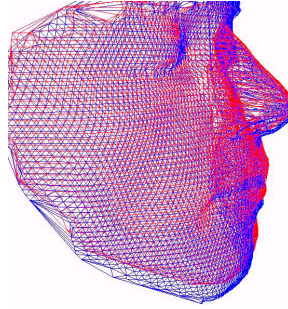
The numerical complexity of one calculation of the measure is important because in applications it is required to calculate the measure several times during iteration process of surface matching. Influence of non-efficient calculation of the measure will increase on each iteration step.

The matching problem, or the problem of spatial alignment of surfaces, consists in transition of several images of the considered object to one global coordinate system (see Fig. 2). Such transition consists in transformation of images using translations along and rotations about the coordinate axes. If we consider transformation parameters that do not derive Delaunay triangulations of the initial meshes from the class of triangulations, the proposed approach allows to implement surface matching in time  $O(mn)$  using  $m$  iterations. In this case we need  $O(n \log n)$  time for preprocessing stage to construct Delaunay triangulations.

## 4 Mesh Localization in Delaunay Triangulation

The well-known geometric search problem of point location in a Delaunay triangulation is formulated as follows: given a point  $Q$  and a Delaunay triangulation  $T$ , it is required to declare the triangle of  $T$  containing  $Q$ . One of the most fast methods to solve this problem (in general case) has  $O(\log N)$  computational complexity and  $O(N)$  memory usage, where  $N$  is the number of nodes in  $T$  [25].

Let  $g$  be a plane mesh of  $N_1$  nodes and  $T_2$  be a Delaunay triangulation constructed on  $N_2$  nodes. In the problem of mesh nodes localization in a Delaunay triangulation it is required to solve point location problem for each node of  $g$ . Then the unstructured mass query of  $N_1$  nodes can be processed by time  $O(N_1 \log N_2)$ .



**Fig. 2.** Matching of facial models

We show how the Delaunay triangulation constructed on the nodes of  $g$  can be used for acquisition of more efficient solution. The proposed method is based on *Euclidean minimum spanning trees* (MST) of given Delaunay triangulation [21] which can be constructed in linear time. There is approach for point location problem that uses "walk along a line" strategy [26]. The idea of the approach consists in gradual transition from some initial point  $M$  of known location to source point  $Q$  along the straight line  $(MQ)$ . During each transition step changing on adjacent (neighboring by edge) triangle is implemented. After such stage is finished, a *location path* consisting of adjacent triangulation triangles is constructed. Case of belonging of a certain node of  $T_2$  to segment  $[MQ]$  is a case of a special interest.

In the proposed method for mesh localization, locations paths pass along the edges of the MST. As a spanning tree does not contain cycles and passes through all points of the mesh  $g$ , the algorithm will work correctly: it does not loop and performs location of all mesh nodes.

In this work we prove that in case of uniform distribution for nodes of triangulation  $T_2$  and mesh  $g$  the average case complexity of the method is linear by  $\max(N_1, N_2)$ . Using the results of Kostuk for average edge length in Delaunay triangulation [27] and of Bose and Devroye for average number of intersections between MST and line segment [28], we show that

**Lemma 1.** *Suppose  $g_1, g_2$  are plane meshes with the numbers of nodes  $N_1, N_2$ , respectively,  $N_1/N_2 \leq c = \text{const}$ , and the sets of nodes of the both meshes uniformly distributed uniformly in a rectangular area. Then, the average number of intersections between the MST for nodes of  $g_1$  and edges of the Delaunay triangulation constructed on  $g_2$  is linear by  $N_2$ .*

**Theorem 1.** *Under the conditions of lemma 1, the algorithm for localization of mesh  $g_1$  in the Delaunay triangulation constructed on the set of nodes of  $g_2$ , on basis of the MST of  $g_1$  has the average case complexity  $O(\max(N_1, N_2))$ .*

The proof of lemma 1 and theorem 1 is omitted.

The assumption of uniform distribution of mesh nodes is appropriate for the majority of practical applications.

In the worst case the mesh localization is not linear. We consider method for construction of simulated example shows that the worst case complexity of the proposed method is quadratic.

In [24] the problem of merging of unseparated Delaunay triangulations were studied: given two Delaunay triangulations  $T_1$  and  $T_2$  constructed on sets  $g_1$  and  $g_2$ , respectively, with intersected convex hulls, it is required to construct the *united Delaunay triangulation*  $T$ . We say that a triangle of  $T$  is an *interface* triangle if it joins nodes from the both sets  $g_1, g_2$ .

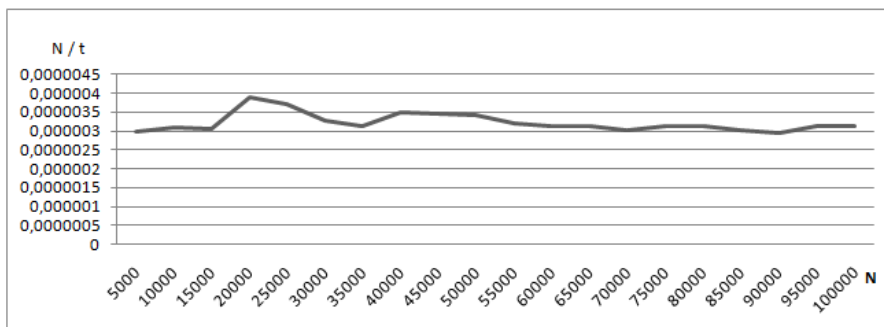
By  $N$  denote the total number of nodes in the sets  $g_1, g_2$ . We consider the problem of identifying of all interface triangles of  $T$ . Using the idea proposed by Mestetskiy and Tsarik for merging of overlapping Delaunay triangulations in [24], we propose the solution of this problem and show that

**Theorem 2.** *All interface triangles of  $T$  can be extracted in linear time by  $N$ .*

The theorem 2 allows to receive the following result:

**Theorem 3.** *Localization of nodes from  $g_1$  in a Delaunay triangulation constructed on  $g_2$  on basis of the list of all interface triangles can be performed in  $O(N)$  in the worst case.*

The theorems 1 and 3 were theoretically proved and experimentally verified. Experiments for estimation of computational complexity were performed on model (see Fig. 3) and real data.



**Fig. 3.** Graph  $N/t — N$ , where  $N \approx N_1 \approx N_2$  — the number of nodes in initial meshes,  $t$  — time for mutual localization of nodes of two Delaunay triangulations

## 5 Applied Problems of 3D Face Image Analysis

As the applications of the proposed methods several problems of 3D face analysis were considered:

- quantitative estimation of facial asymmetry based on comparison of the facial model and the reflected one [29];
- 3D model segmentation on static and dynamic regions by 3D model video sequence on example of chewing process [31]. The idea of estimating the motion parameters of the lower jaw consists in the conditional subdivision of each model of the video sequence into the upper (static) and lower (dynamic) parts and construction for each of the parts of special local coordinate systems of its own. The transformation of the lower system of coordinates into the upper one for every frame of the video sequence describes the dynamics of the lower jaw motion. The formal description of the motion is represented in the form of matrices of the lower coordinates transformation into the upper ones;
- construction of combined spatial model of face and jaws for orthodontics using *reference object*, i.e. object that allows to define geometric relationship of other objects connected with it [32];
- accuracy estimation of 3D model reconstruction methods [33];
- comparison of facial models acquired by scanners of different accuracy [23];
- quantitative estimation of facial skin condition for research in cosmetology [30].

The initial data for mentioned applications were acquired by 3D scanner Broadway designed by Artec Group Company (see [34]). Each face model was normalized in the coordinate system in a special way (transition to a standard coordinate system of a model). Facial models were considered as single-valued surfaces.

The experiments performed on facial model database confirmed the correctness and computational efficiency of the proposed methods.

## 6 Conclusions

Measures for comparing discrete models of surfaces determined at the nodes of different triangulation meshes and the approach for their calculation have been proposed. Several problems of face model analysis were considered as applications of the proposed methods. The proposed approach was theoretically justified and confirmed by multiple computational experiments on 3D face data.

## References

1. Grise, G., Meyer-Hermann, M.: Surface reconstruction using Delaunay triangulation for applications in life sciences. *Computer Physics Communications* 182(4), 967–977 (2011)
2. Szymczak, A., Rossignac, J., King, D.: Piecewise regular meshes: Construction and compression. *Graphical Models* 64(3-4), 183–198 (2002)
3. Gu, X., Gortler, S.J., Hoppe, H.: Geometry images. In: *SIGGRAPH Conference Proceedings*, pp. 355–361 (2002)



4. Alliez, P., Ucelli, G., Gotsman, C., Attene, M.: Recent Advances in Remeshing of Surfaces. In: *Shape Analysis and Structuring: Mathematics and Visualization*, pp. 53–82 (2008)
5. Cignoni, P., Rocchini, C., Scopigno, R.: Metro: measuring error on simplified surfaces. *Computer Graphics Forum* 17(2), 167–174 (1998)
6. Besl, P., McKay, H.: A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14(2), 239–256 (1992)
7. Zhang, Z.: Iterative point matching for registration of freeform curves and surfaces. *International Journal of Computer Vision* 13(2), 119–152 (1994)
8. Fan, T., Medioni, G., Nevatia, R.: Recognizing 3D objects using surface descriptions. *IEEE PAMI* 11(11), 1140–1157 (1989)
9. Chen, Y., Medioni, G.: Object modelling by registration of multiple range images. *Image and Vision Computing* 10(3), 145–155 (1992)
10. Gelfand, N., Ikemoto, L., Rusinkiewicz, S., Levoy, M.: Geometrically Stable Sampling for the ICP Algorithm. In: *Fourth International Conference on 3D Digital Imaging and Modeling*, pp. 260–267 (2003)
11. Friedman, J.H., Bentley, J.L., Finkel, R.A.: An Algorithm for Finding Best Matches in Logarithmic Expected Time. *ACM Transactions on Mathematical Software* 3(3), 209–226 (1977)
12. Liu, Y., Rodrigues, M.A.: Geometrical analysis of two sets of 3D correspondence data patterns for the registration of free-form shapes. *J. Int. and Rob. Systems* 33, 409–436 (2002)
13. Clarkson, K.: A randomized algorithm for closest point queries. *SIAM J. Computing* 17, 830–847 (1998)
14. Brunnstrom, K., Stoddart, A.J.: Genetic algorithms for free-form surface matching. In: *Proc. ICPR*, pp. 689–693 (1996)
15. Godin, G., Rioux, M., Baribeau, R.: Three-dimensional registration using range and intensity information. In: *Proceedings of the SPIE*, vol. 2350, pp. 279–290 (1994)
16. Feldmar, J., Ayache, N., Betting, F.: D-2D projective registration of free-form curves and surfaces. *CVIU* 65, 403–424 (1997)
17. Turk, G., Levoy, M.: Zippered polygon meshes from range images. In: *Proc. SIGGRAPH*, pp. 311–318 (1994)
18. Eggert, D.W., Larusso, A., Fisher, R.B.: Estimating 3-D rigid body transformations: a comparison of four major algorithms. *Machine Vision and Applications* 9(5-6), 272–290 (1997)
19. Schenk, T.: *Digital Photogrammetry*, 428 p. Terra-Science, Laurelville (1999)
20. Vaillant, M., Glaunès, J.: Surface Matching via Currents. In: Christensen, G.E., Sonka, M. (eds.) *IPMI 2005*. LNCS, vol. 3565, pp. 381–392. Springer, Heidelberg (2005)
21. Preparata, F., Shamos, M.: *Computational geometry. An introduction*, 390 p. Springer-Verlag GmbH (1985)
22. Dyshkant, N.: Measures for comparing discrete models of single-valued surfaces. *Moscow University Computational Mathematics and Cybernetics* 35(4), 193–200 (2011)
23. Dyshkant, N.: Measures for Surface Comparison on Unstructured Grids with Different Density. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) *DGCI 2011*. LNCS, vol. 6607, pp. 501–512. Springer, Heidelberg (2011)
24. Mestetskiy, L., Tsarik, E.: Delaunay triangulation: recursion without spatial separation. In: *GrafiCon 2004: Proc. 14th Int. Conf., CMC MSU, Moscow*, pp. 267–270 (2004) (in Russian)

25. Kirkpatrick, D.G.: Optimal search in planar subdivisions. *SIAM J. Comput.* 12(1), 28–35 (1983)
26. Devillers, O., Pion, S., Teillaud, M.: Walking in a triangulation. *Internat. J. Found. Comput. Sci.* 13, 181–199 (2002)
27. Kostuk, Y.L.: Graphic search with the use of triangulation and cellular partition. *Vestnik of Tomsk State University* (275), 147–152 (2002) (in Russian)
28. Bose, P., Devroye, L.: Intersections with Random Geometric Objects. *Computational Geometry: Theory and Applications* 10(3), 139–154 (1998)
29. Dyshkant, N., Mestetskiy, L.: Estimation of Asymmetry in 3D Face Models. In: *International Conference on Computer Vision Theory and Applications (VISAPP 2009)*, pp. 402–405. INSTICC Press, Lisbon (2009)
30. Artuchin, S.: Quantitative Estimation of Skin Condition Based on 3D Face Scanning. In: *18th International Student, Postgraduate and Young Scientist Conference "Lomonosov-2011"*. Maks Press (2011) (in Russian)
31. Gordeev, D., Dyshkant, N.: Construction of Jaw Movement Model During Chewing Process by 3D image sequence. In: *GrafiCon 2009, Proc. 19th Int. Conf. Mosk. Gos. Univ., Moscow*, pp. 348–352 (2009) (in Russian)
32. Dzaraev, C., Persin, L., Porochin, A.: Using of 3D Scanners for Diagnostics of Dentoalveolar Anomalies. In: *Theoretical and Practical Forum Dental-Review, Moscow* (2010) (in Russian)
33. Mestetskiy, L., Dyshkant, N., Gordeev, D., Kumari Sharmila, M., Shekar, B.H.: Surface measures for accuracy evaluation in 3D face reconstruction. *Pattern Recognition* 45(10), 3583–3589 (2012)
34. Artec Group Inc. 3D Scanning Technologies, <http://www.artec3d.com>

# Generalized Simple Surface Points<sup>\*</sup>

J.C. Ciria<sup>1</sup>, E. Domínguez<sup>1</sup>, A.R. Francés<sup>1</sup>, and A. Quintero<sup>2</sup>

<sup>1</sup> Dpto. de Informática e Ingeniería de Sistemas, Facultad de Ciencias,  
Universidad de Zaragoza, E-50009 – Zaragoza, Spain

<sup>2</sup> Dpto. de Geometría y Topología, Facultad de Matemáticas,  
Universidad de Sevilla, Apto. 1160, E-41080 – Sevilla, Spain

**Abstract.** By using exclusively the customary adjacency relations on  $\mathbb{Z}^3$ , we generalize the notion of a simple surface point given by Morgenthaler in the 80s. A new definition of simple surface arises, and we show that simple surfaces coincide with the strong separating family of a certain class of digital surfaces defined by means of continuous analogues that, in turn, contains several families of discrete surfaces in the literature.

## 1 Introduction

It is commonly accepted that the first notion of a discrete surface, as a “thin” subset of voxels of the grid  $\mathbb{Z}^3$ , is due to Morgenthaler and Rosenfeld [8]. They define  $S \subseteq \mathbb{Z}^3$  to be a simple closed surface if all voxels in  $S$  are *simple surface points*, where such points are determined by three properties which are exclusively given in terms of the usual adjacency relations between voxels and subsets of voxels of  $\mathbb{Z}^3$ . Moreover, the notion of a simple surface point is local since it depends only on the 26-neighbourhood of each voxel  $\sigma \in S$ .

Nevertheless, this definition of surface is widely considered to be too restrictive, especially for the (26, 6)-adjacency, since only 13 different configurations, up to rotations and symmetries, may appear in the 26-neighbourhood of a voxel. In fact, relevant classes of discrete surfaces extending the class of Morgenthaler-Rosenfeld surfaces have appeared in the literature since then; as instances, recall the strong surfaces [2] and the simplicity surfaces [6]. Furthermore, these classes are strictly contained in the family of the so-called  $(k, \bar{k})$ -surfaces introduced in [3] (for any adjacency pair  $(k, \bar{k}) \neq (6, 6)$ ,  $k, \bar{k} \in \{6, 18, 26\}$ ) via a plate-based definition. It worth pointing out that (26, 6)-surfaces admit up to 10,580 different configurations in the 26-neighbourhood of a voxel for the (26, 6)-adjacency.

Besides the usual adjacency pairs, any of the classes of discrete surfaces above is defined by a local or global additional notion or property (the definition of a new graph, homotopic properties, . . .). Our aim in this paper is to give a generalized notion of a simple surface point, based solely on the adjacency relations between voxels and sets of voxels, for which the  $(k, \bar{k})$ -surfaces given in [3], in particular the strong and simplicity surfaces, are characterized as sets of simple surface points. This will be achieved as an application of the framework for

---

<sup>\*</sup> This work has been partially supported by the project MTM2010-20445 MICINN Spain.

digital topology in [1]. More precisely, we will use crucially the fact that the  $(k, \bar{k})$ -surfaces in [3] are exactly the strong separating surfaces of certain universal digital space in the sense of [1]; see [5].

## 2 Simple Surface Points

In this section we introduce a new notion of *simple surface point* that extends the original definition of Morgenthaler [8], and find some properties of subsets consisting entirely of voxels of this class. For this we start by recalling some basic definitions of the graph-theoretical approach to the digital topology of  $\mathbb{Z}^3$ .

Two voxels  $\sigma = (x_1^\sigma, x_2^\sigma, x_3^\sigma), \tau = (x_1^\tau, x_2^\tau, x_3^\tau) \in \mathbb{Z}^3$  are said to be 6-, 18- or 26-adjacent if  $\max\{|x_i^\sigma - x_i^\tau|; 1 \leq i \leq 3\} \leq 1$  and they differ in, at most, one, two or three of their coordinates, respectively. The  $k$ -neighbourhood of  $\sigma \in \mathbb{Z}^3$  is the set  $N_k(\sigma)$  of all voxels  $k$ -adjacent to  $\sigma$ . Moreover, we say that two  $n$ -adjacent voxels  $\sigma, \tau$  are *strictly  $n$ -adjacent* if they are not  $m$ -adjacent for any  $m < n$ , where  $n, m \in \{6, 18, 26\}$ . Two voxels  $\sigma, \tau \in X$  are said to be *purely  $n$ -adjacent in  $X$*  ( *$n^X$ -adjacent*, for short) if they are strictly  $n$ -adjacent and no other voxel  $\rho \in X$  is  $m$ -adjacent to both  $\sigma$  and  $\tau$  for  $m < n$ . Finally, two distinct subsets  $X$  and  $Y$  are said to be  $k$ -adjacent if some voxel of  $X$  is  $k$ -adjacent to some voxel of  $Y$ .

For  $k \in \{6, 18, 26\}$  the transitive closure of the  $k$ -adjacency relation defines an equivalence relation on each subset  $X \subseteq \mathbb{Z}^3$ , whose classes are called the  $k$ -components of  $X$ . Subsets with only one  $k$ -component are termed  $k$ -connected. The set  $X$  is said to be  *$k$ -separating* if its complement  $\bar{X} = \mathbb{Z}^3 - X$  consists of two  $k$ -components, and it is a *strong  $k$ -separating* set if each voxel  $\sigma \in X$  is  $k$ -adjacent to both of them. Finally,  $X$  is said to be  *$k$ -thin at a voxel  $\sigma \in X$*  if  $\bar{X}_\sigma = N_{26}(\sigma) - X$  has exactly two  $k$ -components,  $A_\sigma$  and  $B_\sigma$ ,  $k$ -adjacent to  $\sigma$ , and it is simply called a  *$k$ -thin set* if this property holds at each of its voxels.

A *unit cube* of  $\mathbb{Z}^3$  is any subset of eight mutually 26-adjacent voxels. Similarly, a *unit square* of  $\mathbb{Z}^3$  is any subset of four mutually 18-adjacent voxels arising from the intersection of two aligned unit cubes.

Given  $S \subseteq \mathbb{Z}^3$  and an adjacency pair  $(k, \bar{k}), k, \bar{k} \in \{6, 18, 26\}$ , Morgenthaler's definition [8] states that  $\sigma \in S$  is a (simple) surface point if the following conditions hold: (a) exactly one  $k$ -component of  $N_{26}(\sigma) \cap S - \{\sigma\}$  is  $k$ -adjacent to  $\sigma$ ; (b)  $S$  is  $\bar{k}$ -thin at  $\sigma$ ; and, (c) every voxel  $\tau \in S$   $k$ -adjacent to  $\sigma$  is  $\bar{k}$ -adjacent to the  $\bar{k}$ -components  $A_\sigma$  and  $B_\sigma$  of  $\bar{S}_\sigma$ . In fact, Kong and Roscoe [7] showed that property (a) is redundant for all sets consisting entirely of surface points.

As announced in the introduction we aim at weakening the original definition of Morgenthaler and Rosenfeld to characterize the remarkably large class of  $(k, \bar{k})$ -surfaces in [3]. Since it seems quite reasonable to keep property (b) to guarantee suitable separation properties, our goal will be attained by finding a less restrictive version of property (c). For this, notice that if  $S \subseteq \mathbb{Z}^3$  is  $\bar{k}$ -thin and  $\sigma \in S$  is a surface point then, for each voxel  $\tau \in S$   $k$ -adjacent to  $\sigma$ , property (c) implies that  $A_\sigma \cap (A_\tau \cup B_\tau) \neq \emptyset \neq B_\sigma \cap (A_\tau \cup B_\tau)$ . This fact suggests the following preliminary definition.

**Definition 1.** Let  $O \subseteq \mathbb{Z}^3$  be  $\bar{k}$ -thin at two 26-adjacent voxels  $\sigma, \tau \in O$ . We say that  $\sigma$  is  $\bar{k}_6$ -linked to  $\tau$  in  $O$  if the  $\bar{k}$ -components  $A_\sigma$  and  $B_\sigma$  of  $\bar{O}_\sigma$  meet  $A_\tau \cup B_\tau$ . A sequence  $\sigma' = \rho_0, \rho_1, \dots, \rho_n = \tau'$  of voxels such that  $\rho_{i-1}$  is  $\bar{k}_6$ -linked to  $\rho_i$  in  $O$ , for  $1 \leq i \leq n$ , will be termed a  $\bar{k}_6$ -path from  $\sigma'$  to  $\tau'$  in  $O$ .

*Remark 2.* If  $O \subseteq \mathbb{Z}^3$  is both  $\bar{k}$ -thin and strong  $\bar{k}$ -separating then, for each voxel  $\sigma \in O$ , the  $\bar{k}$ -components  $A_\sigma$  and  $B_\sigma$  of  $\bar{O}_\sigma$  are representing the  $\bar{k}$ -components,  $A$  and  $B$ , of  $\mathbb{Z}^3 - O$ . Therefore, if  $\sigma$  is  $\bar{k}_6$ -linked to  $\tau$  then  $A_\sigma \cap B_\tau = \emptyset = B_\sigma \cap A_\tau$  and, thus,  $A_\sigma \cap A_\tau \neq \emptyset \neq B_\sigma \cap B_\tau$ . Moreover, if  $(\sigma_i)_{i=0}^n$  is a  $\bar{k}_6$ -path in  $O$  then  $\cup_{i=1}^n A_{\sigma_i} \subseteq A$  and  $\cup_{i=1}^n B_{\sigma_i} \subseteq B$  are  $\bar{k}$ -connected.

**Lemma 3.** If  $\sigma \in O$  is  $\bar{k}_6$ -linked to  $\tau$  then  $\bar{O}_\sigma \cap \bar{O}_\tau$  contains voxels of  $A_\sigma$  and  $B_\sigma$  and thus it is not  $\bar{k}$ -connected. In particular,  $\sigma$  and  $\tau$  are not  $\bar{k}_6$ -linked if they are  $\bar{k}^O$ -adjacent for  $\bar{k} \in \{18, 26\}$ .

Any surface point  $\sigma \in O$  in the sense of Morgenthaler is trivially  $\bar{k}_6$ -linked to each of its  $k$ -neighbours provided  $O$  is also  $\bar{k}$ -thin at them. Next result is a partial converse for  $\bar{k} \in \{18, 26\}$ .

**Proposition 4.** Assume  $O$  is  $\bar{k}$ -thin at two 26-adjacent voxels  $\sigma, \tau \in O$ ,  $\bar{k} \in \{18, 26\}$ . Then  $\sigma$  is  $\bar{k}_6$ -linked to  $\tau$  if and only if  $\tau$  is  $\bar{k}$ -adjacent to  $A_\sigma$  and  $B_\sigma$ .

*Proof.* Assume the  $\bar{k}_6$ -linking of  $\sigma$  and  $\tau$ , and take  $\rho \in A_\sigma \cap (A_\tau \cup B_\tau)$ . The result is obvious if  $\rho$  is  $\bar{k}$ -adjacent to  $\tau$ , in particular for  $\bar{k} = 26$  since  $A_\tau \cup B_\tau \subseteq N_{26}(\tau)$ . So, assume  $\bar{k} = 18$  and  $\rho$  is strictly 26-adjacent to  $\tau$ , and let  $K_{\rho\tau}$  the only unit cube of  $\mathbb{Z}^3$  containing  $\{\rho, \tau\}$ . Since  $\rho \in A_\tau$ , there exists another voxel  $\rho' \in K_{\rho\tau} - (O \cup \{\rho\})$  which is 18-adjacent to both  $\rho$  and  $\tau$ ; in particular  $K_{\rho\tau} - O$  is 18-connected. On the other hand,  $\sigma$  also belongs to  $K_{\rho\tau}$  since it is 26-adjacent to  $\tau$  and  $\rho$ . Moreover,  $\rho$  is necessarily 18-adjacent to  $\sigma$  and the result follows since  $\rho' \in K_{\rho\tau} - O \subseteq A_\sigma$ . The converse is obvious since any voxel  $\rho \in A_\sigma$   $\bar{k}$ -adjacent to  $\tau$  is necessarily in  $A_\tau \cup B_\tau$ .  $\square$

*Example 5.* Although weaker than property (c) of Morgenthaler’s surface points, the  $\bar{k}_6$ -linking requirement in Def. 1 is still too restrictive for adjacency pairs  $(k, \bar{k})$  with  $k \in \{18, 26\}$ . For example, let us consider the subsets  $O_1, O_2, O_3 \subseteq \mathbb{Z}^3$  shown in Fig. 1, which are pieces of a simplicity 26-surface  $S_1$ , a  $(26, 6)$ -surface  $S_2$  in the sense of [3] (but no simplicity 26-surface contains  $O_2$ ) and a pinched torus  $S_3$ , respectively. Notice that, in all of them, the 26-adjacent voxels  $\sigma$  and  $\tau$  are not  $\bar{k}_6$ -linked to each other for  $(k, \bar{k}) = (26, 6)$ , while the sequences  $(\sigma, \rho, \tau)$  and  $(\sigma, \rho_2, \rho_1, \tau)$  are  $\bar{k}_6$ -paths in  $O_1$  and  $O_2$ ; that is, according to the following definition,  $\sigma$  and  $\tau$  are  $\bar{k}_{18}$ -linked in  $O_1$  and  $\bar{k}_{26}$ -linked in  $O_2$ . In contrast with  $S_1$  and  $S_2$ , despite there exist  $\bar{k}_6$ -paths from  $\sigma$  to  $\tau$  in the pinched torus  $S_3$ , they can not be found within the two unit cubes containing them, and so they are not  $\bar{k}_{26}$ -linked in  $O_3$ .

**Definition 6.** A voxel  $\sigma \in O$  is  $\bar{k}_{18}$ -linked to  $\tau$  in  $O$  if there exists a unit cube  $K \subseteq \mathbb{Z}^3$  containing a  $\bar{k}_6$ -path from  $\sigma$  to  $\tau$  in  $O$ ; while  $\sigma$  is  $\bar{k}_{26}$ -linked to  $\tau$  in  $O$  if such a  $\bar{k}_6$ -path is found in the union of two unit cubes that meet in a unit square.

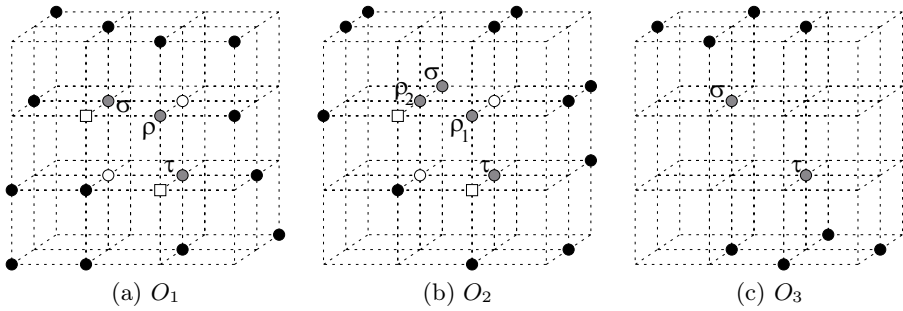


Fig. 1. See Example 5

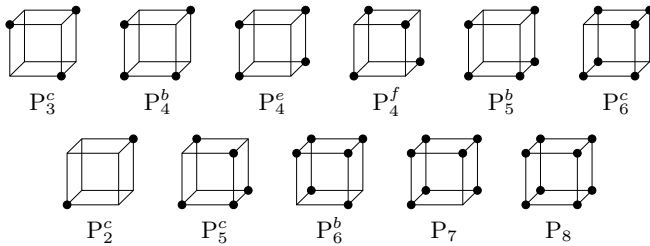


Fig. 2. Some of the patterns that may appear in any digital object  $O \subseteq \mathbb{Z}^3$ . Each picture represents a unit cube  $K \subseteq \mathbb{Z}^3$ , and the black dots are the set of voxels in  $O \cap K$ . The patterns in the lower row cannot appear in  $O$  if it consists entirely of simple  $(26, 6)$ -surface points.

**Definition 7.** A voxel  $\sigma \in S$  is a simple  $(k, \bar{k})$ -surface point if the two following conditions are satisfied:

1.  $S$  is  $\bar{k}$ -thin at  $\sigma$ , and also at each voxel  $\tau \in S$   $k$ -adjacent to  $\sigma$ .
2. For every voxel  $\tau \in N_k(\sigma) \cap S$ ,  $\sigma$  is  $\bar{k}_t$ -linked to  $\tau$  in  $O$  whenever  $\sigma$  and  $\tau$  are strictly  $t$ -adjacent, where  $t \in \{6, 18, 26\}$  and  $t \leq k$ .

This new notion of simple surface point leads to the following definition.

**Definition 8.** A simple closed  $(k, \bar{k})$ -surface is a subset  $S \subseteq \mathbb{Z}^3$  consisting entirely of simple  $(k, \bar{k})$ -surface points such that if  $K - S$  is not  $\bar{k}$ -connected, for a given unit cube  $K$ , then it meets  $A_\sigma$  and  $B_\sigma$  for each voxel  $\sigma \in K \cap S$ .

As a preparatory work for the main result of the paper (Th. 19), we next prove that some of the 22 non-empty patterns that, up to rotations and symmetries, may appear in the intersection  $K \cap O$  of a unit cube  $K$  and an arbitrary subset  $O$  are not allowed in objects consisting entirely of simple  $(k, \bar{k})$ -surface points (see Fig. 2). The proof of the next proposition requires a tedious analysis of cases that we skip in order to keep the length of the paper within the required size.

**Proposition 9.** Let  $O \subseteq \mathbb{Z}^3$  be a  $\bar{k}$ -thin set such that any pair of 6-adjacent voxels  $\sigma, \tau \in O$  are  $\bar{k}_6$ -linked to each other. Then  $K \cap O \notin \mathbb{FC}_{\bar{k}}$  for each unit cube  $K$  of  $\mathbb{Z}^3$ , where  $\mathbb{FC}_6 = \{P_5^c, P_6^b, P_7, P_8\}$ ,  $\mathbb{FC}_{18} = \{P_7, P_8\}$  and  $\mathbb{FC}_{26} = \{P_8\}$ .

**Proposition 10.** *Let  $\sigma, \tau \in O$  be two  $26^O$ -adjacent voxels. Then  $\sigma$  and  $\tau$  are not  $\bar{k}_{26}$ -linked for any  $\bar{k} \in \{6, 18, 26\}$ .*

*Proof.* Let  $K \subseteq \mathbb{Z}^3$  the only unit cube containing  $\{\sigma, \tau\}$ . Notice that  $K \cap O = \{\sigma, \tau\}$  and thus  $K - O$  is 6-connected. Therefore these voxels are not  $\bar{k}_6$ -linked by Lemma 3. And the result follows since  $\sigma$  and  $\tau$  are consecutive elements in any 26-path contained in  $(K \cup K') \cap O$ , where  $K'$  is another unit cube of  $\mathbb{Z}^3$ .  $\square$

### 3 Universal $(k, \bar{k})$ -Spaces and Digital Surfaces

This section collects the terminology and main results from [5] needed in this paper. They are established within the framework for Digital Topology introduced in [1], where we refer to for the precise definitions in a general setting. The reader should be aware that some of the material referred to [5] and [1] has been restated here in terms of the notation introduced in Section 2.

In this paper we will only deal with the polyhedral complex  $R^3$  determined by the collection of unit cubical cells in the Euclidean space  $\mathbb{R}^3$  centred at points of integer coordinates. More precisely, let  $\mathcal{Z} = \{z + \frac{1}{2}; z \in \mathbb{Z}\}$  and  $\mathcal{L} = \{[l, l+1]; l \in \mathbb{Z}\}$ , then a (cubical)  $d$ -cell  $\delta \in R^3$ ,  $0 \leq d \leq 3$ , is the cartesian product of  $d$  elements of  $\mathcal{L}$  and  $3 - d$  elements of  $\mathcal{Z}$ . As usual, if  $\delta \in R^3$  is a  $d$ -cell we say that  $d$  is the *dimension* of  $\delta$  and write  $\dim \delta = d$ . Given two cells  $\alpha, \beta \in R^3$  we write  $\alpha \leq \beta$  if  $\alpha$  is a *face* of  $\beta$  (i.e., if  $\alpha \subseteq \beta$ ), and  $\alpha < \beta$  if in addition  $\alpha \neq \beta$ .

Notice that each  $d$ -cell  $\delta \in R^3$  can be associated to its centre  $c(\delta)$  which, in particular, belongs to  $\mathbb{Z}^3$  whenever  $\dim \delta = 3$ . This way, the complex  $R^3$  can be regarded as a dual representation of the discrete space provided by  $\mathbb{Z}^3$ , in which each 3-cell represents a voxel, and so any subset of  $\mathbb{Z}^3$  is identified with a subset of the set  $\text{cell}_3(R^3)$  of 3-cells in  $R^3$ . This identification, that will be used henceforth without further comment, gives us a one-to-one correspondence between 0-cells (1-cells) and discrete unit cubes (squares, respectively) of  $\mathbb{Z}^3$ . Namely, for each 0-cell (1-cell)  $\alpha \in R^3$  the set of voxels  $K_\alpha = \{\sigma \in \text{cell}_3(R^3); \alpha \leq \sigma\}$  is the unit cube (square, respectively) centred at  $c(\alpha)$ . Moreover, if  $\dim \alpha = 2$  then  $K_\alpha = \{\sigma, \tau\}$ , where  $\sigma$  and  $\tau$  are 6-adjacent and  $\alpha = \sigma \cap \tau$ , while  $K_\alpha = \{\alpha\}$  if  $\dim \alpha = 3$ . In any case, we will refer to  $\alpha$  as the *centre* of  $K_\alpha$ .

According to [1] a *digital space* based on  $R^3$  is a pair  $(R^3, f)$  where  $f$  is a *lighting function*: a certain map of the form  $f : \mathcal{P}(\text{cell}_3(R^3)) \times R^3 \rightarrow \{0, 1\}$  satisfying five axioms, where  $\mathcal{P}(\text{cell}_3(R^3))$  stands for the family of all subsets of  $\text{cell}_3(R^3)$ . Here we simply recall that lighting functions are devised as procedures for associating a continuous analogue to a digital object. More precisely, given a lighting function  $f$  as above, the *continuous analogue* of an object  $O \subseteq \text{cell}_3(R^3)$  is the polyhedron  $|\mathcal{A}_O^f| \subseteq \mathbb{R}^3$  triangulated by the *simplicial complex*<sup>1</sup>  $\mathcal{A}_O^f$  consisting of all simplexes  $\langle c(\sigma_0), c(\sigma_1), \dots, c(\sigma_n) \rangle$  whose vertices are centres  $c(\sigma_i)$  of cells  $\sigma_0 < \sigma_1 < \dots < \sigma_n \in R^3$  such that  $f(O, \sigma_i) = 1$  (that is,  $f$  “lights”  $\sigma_i$  in  $O$ ).

---

<sup>1</sup> We will drop the “ $f$ ” from the notation and write  $\mathcal{A}_O$  instead  $\mathcal{A}_O^f$  whenever the lighting function involved is clear from the context. Also we write  $\mathcal{A}_{R^3}$  instead  $\mathcal{A}_{\text{cell}_3(R^3)}$ .

Regarding continuous analogues as “continuous interpretations” of digital images, we introduce digital notions in terms of the corresponding continuous ones. In particular, an object  $S \subseteq \text{cell}_3(\mathbb{R}^3)$  is a *digital surface* in  $(\mathbb{R}^3, f)$  if its continuous analogue  $|\mathcal{A}_S|$  is a combinatorial surface; that is, if the link  $\text{lk}(v; \mathcal{A}_S) = \{A \in \mathcal{A}_S; v, A < B \in \mathcal{A}_S \text{ and } v \notin A\}$  is a 1-sphere for each vertex  $v \in \mathcal{A}_S$ . The scope of this paper is restricted to surfaces in digital spaces for which the connectedness of the continuous analogues is coherent with the connectedness induced by an adjacency pair  $(k, \bar{k})$  on the grid  $\mathbb{Z}^3$ . More precisely,  $(\mathbb{R}^3, f)$  is said to be a  $(k, \bar{k})$ -space if the two following conditions hold for any subset  $O \subseteq \mathbb{Z}^3$ : (a)  $O$  is  $k$ -connected if and only if  $|\mathcal{A}_O|$  is a connected polyhedron; and (b) the complement  $\mathbb{Z}^3 - O$  is  $\bar{k}$ -connected if and only if  $|\mathcal{A}_{\mathbb{R}^3}| - |\mathcal{A}_O|$  is connected. As a consequence of the well-known Jordan-Brouwer theorem one gets the following separation result; see [5].

**Theorem 11 (Th. 2 in [5]).** *Let  $(\mathbb{R}^3, f)$  be a  $(k, \bar{k})$ -space such that  $|\mathcal{A}_{\mathbb{R}^3}| = \mathbb{R}^3$ . Then, each  $k$ -connected digital surface  $S$  in  $(\mathbb{R}^3, f)$  is a  $\bar{k}$ -separating set.*

Several different examples of  $(k, \bar{k})$ -spaces for each adjacency pair can be found in [3] and its references. Nevertheless, one particular among them, the so-called *universal  $(k, \bar{k})$ -space*,  $(\mathbb{R}^3, f_{k\bar{k}})$ , captures the essence of all digital surfaces for the pair  $(k, \bar{k})$ . More precisely, the main result in [5] states that any digital surface in a  $(k, \bar{k})$ -space is also a digital surface in  $(\mathbb{R}^3, f_{k\bar{k}})$ . Next we recall the definition of the lighting functions  $f_{k\bar{k}}$  and some immediate properties.

**Definition 12.** *For  $(k, \bar{k}) \neq (6, 6)$ ,  $k, \bar{k} \in \{6, 18, 26\}$ , the lighting functions  $f_{k\bar{k}}$  are defined as follows. Given  $O \subseteq \text{cell}_3(\mathbb{R}^3)$  and a cell  $\delta \in \mathbb{R}^3$ ,  $f_{k\bar{k}}(O, \delta) = 1$  if either  $K_\delta \subseteq O$  or one of the following conditions holds:*

1.  $\dim \delta = 0$  and  $K_\delta \cap O$  corresponds (up to rotations and symmetries) to some pattern in the set  $\mathbb{P}_{\bar{k}} \cup \mathbb{FP}_{k\bar{k}}$ , where  $\mathbb{P}_6 = \{\mathbb{P}_3^c, \mathbb{P}_4^b, \mathbb{P}_4^e, \mathbb{P}_4^f, \mathbb{P}_5^b, \mathbb{P}_6^c\}$ ,  $\mathbb{P}_{18} = \{\mathbb{P}_6^c\}$ ,  $\mathbb{P}_{26} = \emptyset$  and  $\mathbb{FP}_{26,6} = \{\mathbb{P}_2^c, \mathbb{P}_5^c, \mathbb{P}_6^b, \mathbb{P}_8\}$ ,  $\mathbb{FP}_{26,\bar{k}} = \{\mathbb{P}_2^c, \mathbb{P}_8\}$  if  $\bar{k} \neq 6$ ,  $\mathbb{FP}_{18,6} = \{\mathbb{P}_5^c, \mathbb{P}_6^b, \mathbb{P}_8\}$ , and  $\mathbb{FP}_{k\bar{k}} = \{\mathbb{P}_8\}$  otherwise; see Fig. 2
2.  $\dim \delta = 1$ ,  $K_\delta \cap O = \{\sigma, \tau\}$  consists of two  $18^O$ -adjacent voxels, and one of the next further conditions also holds: (a) for  $\bar{k} = 6$ , and  $k \neq 6$ ,  $f_{k,6}(O, \alpha_1) = f_{k,6}(O, \alpha_2)$ , where  $\alpha_1, \alpha_2$  are the two vertices of the 1-cell  $\delta$ ; or (b)  $\sigma$  and  $\tau$  belong to distinct 6-components of  $(K_{\alpha_1} \cup K_{\alpha_2}) \cap O$ , for  $k, \bar{k} \in \{18, 26\}$ .

*Remark 13.* The non-empty difference  $K - O$  of a unit cube  $K$  and  $O \subseteq \mathbb{Z}^3$  is trivially 26-connected. Moreover, it is 18-connected if and only if  $K \cap O$  does not correspond to the pattern  $\mathbb{P}_6^c$ , and it is 6-connected if and only if  $K \cap O \notin \mathbb{P}_6 \cup \{\mathbb{P}_5^c, \mathbb{P}_6^b\}$ . Moreover, it is easily checked that  $K - O$  has two 6-components whenever  $K \cap O \in \mathbb{P}_6 - \{\mathbb{P}_4^f\}$ .

**Lemma 14.** *If  $\sigma \in O$  and  $\tau \notin O$  are  $\bar{k}$ -adjacent for a given set  $O \subseteq \text{cell}_3(\mathbb{R}^3)$ , then  $f_{k\bar{k}}(O, \sigma \cap \tau) = 0$ .*

For the universal  $(k, \bar{k})$ -spaces, the Jordan-Brouwer theorem also yields the next characterization of the two  $\bar{k}$ -components of  $\text{cell}_3(\mathbb{R}^3) - S$  in terms of the difference of polyhedra  $D_\delta = |\text{st}(c(\delta), \mathcal{A}_{\mathbb{R}^3})| - |\text{st}(c(\delta), \mathcal{A}_S)|$ , where  $\delta \in \mathbb{R}^3$  is any



cell for which  $f_{\bar{k}}(S, \delta) = 1$  and  $\text{st}(c(\delta), \mathcal{A}_X) = \{A \in \mathcal{A}_X; c(\delta), A < B \in \mathcal{A}_X\}$  is the *star* of  $c(\delta)$  in the complex  $\mathcal{A}_X$ ; see [1, Appendix A] for details. This result will be crucial in showing that strong  $\bar{k}$ -separation is equivalent to  $\bar{k}$ -thinness for digital surfaces in  $(R^3, f_{\bar{k}})$ .

**Theorem 15.** *Let  $S$  be a  $k$ -connected digital surface in  $(R^3, f_{\bar{k}})$  and let  $\delta \in R^3$  be a cell for which  $f_{\bar{k}}(S, \delta) = 1$ . Moreover, assume the voxels  $\sigma_1, \sigma_2 \notin S$  have a face  $\alpha_i < \sigma_i$  such that  $c(\alpha_i) \in D_\delta$ ,  $i = 1, 2$ . Then  $\sigma_1$  and  $\sigma_2$  belong to the same  $\bar{k}$ -component of  $\text{cell}_3(R^3) - S$  if and only if  $c(\alpha_1)$  and  $c(\alpha_2)$  are in a connected component of  $D_\delta$ . In particular,  $D_\delta$  have two connected components.*

**Lemma 16.** *Let  $\sigma \in O$ . If  $f_{\bar{k}}(O, \delta) = 0$  for a face  $\delta < \sigma$  then  $K_\delta - O \neq \emptyset$  and it is contained in a  $\bar{k}$ -component of  $\bar{O}_\sigma = N_{26}(\sigma) - O$ .*

*Proof.* Firstly notice that  $\delta \leq \sigma \cap \tau$  for any voxel  $\tau \in K_\delta$ , so that  $\sigma$  and  $\tau$  are 26-adjacent, and, moreover,  $K_\delta \not\subseteq O$  by Def. 12. Then, if  $\dim \delta = 2$ ,  $K_\delta = \{\sigma, \tau\}$  and  $K_\delta - O = \{\tau\}$  is trivially  $\bar{k}$ -connected. In case  $\dim \delta = 0$  the result follows from Remark 13 since  $K_\delta \cap O \notin \mathbb{P}_{\bar{k}} \cup \mathbb{F}_{\bar{k}}$  by Def. 12. Finally, for  $\dim \delta = 1$  the result is obvious if  $\bar{k} \in \{18, 26\}$  since  $K_\delta$  is a unit square. So, assume  $\bar{k} = 6$  and, moreover,  $K_\delta - O$  is not 6-connected or, equivalently  $K_\delta \cap O = \{\sigma, \tau\}$  consists of two 18<sup>O</sup>-adjacent voxels. Given that  $f_{k6}(O, \delta) = 0$ , Def. 12 yields that  $f_{k6}(O, \alpha) = 0$  for some vertex  $\alpha < \delta$ , and then  $K_\delta - O \subseteq K_\alpha - O$  which has already been proved to be  $\bar{k}$ -connected. □

**Proposition 17.** *Let  $\sigma \in O$  be a voxel in an arbitrary set. Then, the set  $\cup_{c(\delta) \in X} K_\delta - O$  is contained in a  $\bar{k}$ -component of  $\bar{O}_\sigma$  for each component  $X$  of  $D_\sigma = |\text{st}(c(\sigma), \mathcal{A}_{R^3})| - |\text{st}(c(\sigma), \mathcal{A}_O)|$ .*

*Proof.* Given  $c(\delta) \in X$  we have  $f_{\bar{k}}(O, \delta) = 0$ ; hence  $K_\delta - O$  is contained in a  $\bar{k}$ -component,  $A_\delta$ , of  $\bar{O}_\sigma$  by Lemma 16. It will suffice to check that  $A_\delta = A_{\delta'}$  for any two cells  $\delta, \delta' \in R^3$  such that  $c(\delta), c(\delta') \in X$ .

Indeed, there exists a sequence  $(\delta_i)_{i=0}^n$  of faces of  $\sigma$  whose centres connect  $c(\delta)$  and  $c(\delta')$  in  $X$ ; that is,  $\delta_0 = \delta$ ,  $\delta_n = \delta'$  and either  $\delta_i < \delta_{i+1}$  or  $\delta_i > \delta_{i+1}$  for each  $1 \leq i \leq n$ . By using an inductive argument we will show that  $B_i = \cup_{j=0}^i K_{\delta_j} - O \subseteq A_\delta$  for  $0 \leq i \leq n$ , and the result follows since  $K_{\delta'} - O \subseteq B_n$ . For  $i = 0$  the result is obvious. Then, assume  $B_i \subseteq A_\delta$  for some index  $i < n$ . If  $\delta_i < \delta_{i+1}$  then  $K_{\delta_{i+1}} \subseteq K_{\delta_i}$  and, henceforth,  $B_{i+1} = B_i$ . On the other hand, if  $\delta_{i+1} < \delta_i$  then  $\emptyset \neq K_{\delta_i} - O \subseteq B_i \cap (K_{\delta_{i+1}} - O)$ . Therefore  $B_i, K_{\delta_{i+1}} - O$  and their union  $B_{i+1}$  are contained in  $A_\delta$ . □

**Theorem 18.** *A  $k$ -connected digital surface  $S$  in  $(R^3, f_{\bar{k}})$  is strong  $\bar{k}$ -separating if and only if it is a  $\bar{k}$ -thin object.*

*Proof.* Assume  $\tau_1, \tau_2 \notin S$  are voxels  $\bar{k}$ -adjacent to a given voxel  $\sigma \in S$ , and let  $\alpha_i = \tau_i \cap \sigma$ ,  $i = 1, 2$ . By Lemma 14 and Th. 15 we know that  $\tau_1$  and  $\tau_2$  are in the same  $\bar{k}$ -component of  $\text{cell}_3(R^3) - S$  if and only if  $\{c(\alpha_1), c(\alpha_2)\}$  is contained in a component of  $D_\sigma$  and, by Prop. 17, this occurs if and only if  $\tau_1$  and  $\tau_2$  are also in a  $\bar{k}$ -component of  $\bar{S}_\sigma$ . □

## 4 Main Theorem

In this section we state and prove our main result (Th. 19 below). Doing that we gain a third equivalent approach to the class  $\mathcal{S}_{k\bar{k}}$  of strong separating digital surfaces in the universal space  $(R^3, f_{k\bar{k}})$ . From each approach distinct facets of the family  $\mathcal{S}_{k\bar{k}}$  are revealed. Namely, the original definition, based on continuous analogues, allows to show that, among others,  $\mathcal{S}_{26,6}$  strictly contains the classes of strong 26-surfaces and simplicity 26-surfaces [1,4]. The equivalent plate-based definition of a  $(k, \bar{k})$ -surface introduced in [3] provides a geometrical interpretation of the elements of  $\mathcal{S}_{k\bar{k}}$  as well as a tool to compute the number of different configurations that they may contain in the 26-neighbourhood of a voxel: up to 10,580 for  $\mathcal{S}_{26,6}$ . Nevertheless being the latter a local definition, with plates resembling patches of the surface within unit cubes, it requires an auxiliary graph for describing how these plates are glued to each other. As an advantage, the new equivalence proved in Th. 19 in terms of simple  $(k, \bar{k})$ -surfaces relies solely on the adjacency relations between voxels and subsets of voxels.

**Theorem 19.** *For  $(k, \bar{k}) \neq (6, 6)$ ,  $k, \bar{k} \in \{6, 18, 26\}$ , a subset  $S \subseteq \mathbb{Z}^3$  is a  $k$ -connected simple  $(k, \bar{k})$ -surface if and only if it is a strong  $\bar{k}$ -separating  $k$ -connected digital surface in  $(R^3, f_{k\bar{k}})$ .*

Since the proofs of the two parts of Th. 19 use techniques quite different from each other and they are, in addition, rather long, we have divided this section into two subsections, one for each part of the proof.

### 4.1 Proof of the “if” Part of Theorem 19

Recall that if  $S$  is a  $k$ -connected strong  $\bar{k}$ -separating digital surface in the universal  $(k, \bar{k})$ -space  $(R^3, f_{k\bar{k}})$ ,  $(k, \bar{k}) \neq (6, 6)$  and  $k, \bar{k} \in \{6, 18, 26\}$ , then it is a  $\bar{k}$ -thin object by Th. 18. Moreover, given a voxel  $\sigma \in S$ , Th. 15 yields that  $D_\sigma = |\text{st}(c(\delta), \mathcal{A}_{R^3})| - |\text{st}(c(\delta), \mathcal{A}_S)|$  has two connected components,  $C_A, C_B$ , that determine the two  $\bar{k}$ -components of  $\overline{S}_\sigma = N_{26}(\sigma) - S$  which are  $\bar{k}$ -adjacent to  $\sigma$ . More precisely, from Th. 18 and Prop. 17 one gets  $\cup_{c(\delta) \in C_X} K_\delta - S \subseteq X_\sigma$ ,  $X \in \{A, B\}$ . After these observations we are ready to prove the next proposition.

**Proposition 20.** *Let  $K_\alpha \subseteq \mathbb{Z}^3$  be a unit cube. Then, for each voxel  $\sigma \in K_\alpha \cap S$ ,  $K_\alpha - S$  meets  $A_\sigma$  and  $B_\sigma$  whenever it is not  $\bar{k}$ -connected.*

*Proof.* If  $K_\alpha - S \neq \emptyset$  is not  $\bar{k}$ -connected then  $f_{k\bar{k}}(S, \alpha) = 1$ , and  $\bar{k} = 6, 18$ , by Remark 13 and Def. 12. Then, Th. 15 yields that  $D_\alpha$  has two connected components,  $X_1, X_2$ , and the sets  $C_i = \cup_{c(\delta) \in X_i} K_\delta - S$  are in distinct  $\bar{k}$ -components of  $\mathbb{Z}^3 - S$ . Hence, for each voxel  $\sigma \in K_\alpha \cap S$ ,  $C_1$  and  $C_2$  are in distinct  $\bar{k}$ -components of  $\overline{O}_\sigma$ . And the result follows from the observations above since an immediate checking shows that there exists a voxel  $\sigma_i \in C_i$  having a face  $\delta_i \leq \sigma \cap \sigma_i$  such that  $f_{k\bar{k}}(S, \delta_i) = 0$ .  $i = 1, 2$ .  $\square$

After this result we will be done if we show that each voxel in  $S$  is a simple  $(k, \bar{k})$ -surface point. For this it will be enough to prove the following.

**Proposition 21.** *Let  $\sigma, \tau \in S$  be two  $k$ -adjacent voxels which are strictly  $t$ -adjacent, where  $t \leq k$  and  $t \in \{6, 18, 26\}$ . Then  $\sigma$  is  $\bar{k}_t$ -linked to  $\tau$ .*

*Proof.* Let  $\delta = \sigma \cap \tau$ . If  $\sigma$  and  $\tau$  are 6-adjacent, then  $\dim \delta = 2$  and  $f_{k\bar{k}}(S, \delta) = 1$  by Def. 12. Since any cell  $\alpha < \delta$  is also a face of  $\sigma$  and  $\tau$  we have that  $D_\delta \subseteq D_\sigma \cap D_\tau$ . Therefore, the observations above yield that  $\cup_{c(\alpha) \in C_X} K_\alpha - S \subseteq X_\sigma \cap X_\tau$ ,  $X \in \{A, B\}$ , where  $C_A$  and  $C_B$  are the two components of  $D_\delta$ . Hence  $\sigma$  and  $\tau$  are  $\bar{k}_6$ -linked.

Assume  $k = 18, 26$  and  $\sigma, \tau$  are strictly 18-adjacent voxels. It is obvious that they are  $\bar{k}_{18}$ -linked if we find a third voxel  $\rho \in S$  6-adjacent to both, since we have already proved that each pair of 6-adjacent voxels in  $S$  are  $\bar{k}_6$ -linked. So, assume  $\sigma, \tau$  are 18<sup>S</sup>-adjacent and let  $\{\sigma', \tau'\} = K_\delta - S$ . If  $\bar{k} = 6$  and  $f_{k\bar{k}}(S, \delta) = 1$  then it is readily checked that  $\{c(\sigma')\}$  and  $\{c(\tau')\}$  are the two components of  $D_\delta$ . Therefore,  $\sigma$  and  $\tau$  are  $\bar{k}_6$ -linked since the observations above yield  $\sigma' \in A_\sigma \cap A_\tau$  and  $\tau' \in B_\sigma \cap B_\tau$ . On the other hand, if  $f_{k\bar{k}}(S, \delta) = 0$  then  $f_{k\bar{k}}(S, \alpha) = 1$  for exactly one of the two vertices  $\alpha < \delta$ . Then it is not difficult to find a  $\bar{k}_6$ -path in  $K_\alpha \cap S$ , which actually corresponds to pattern  $P_4^f$ , from  $\sigma$  to  $\tau$ . Finally, if  $\bar{k} = 18, 26$ ,  $f_{k\bar{k}}(S, \alpha_i) = 0$  for the two vertices  $\alpha_i$  of  $\delta$ . Hence  $f_{k\bar{k}}(S, \delta) = 0$  and Def. 12 yields a 6-path, from  $\sigma$  to  $\tau$  in  $(K_{\alpha_1} \cup K_{\alpha_2}) \cap S$ . Notice that  $f_{k\bar{k}}(S, \delta) = 1$  is ruled out, since then  $\text{lk}(c(\delta), \mathcal{A}_S) = \{c(\sigma), c(\tau)\}$  is not a 1-sphere.

Finally, if  $k = 26$  and  $\sigma, \tau$  are strictly 26-adjacent then there exists at least one more voxel in  $K_\delta \cap S$  and the result follows by the previous cases. Notice that  $K_\delta \cap S = P_2^c$  can not occur since, otherwise  $f_{26, \bar{k}}(S, \delta) = 1$  and  $\text{lk}(c(\delta), \mathcal{A}_S) = \{c(\sigma), c(\tau)\}$  is not a 1-sphere and so  $S$  is not a digital surface.  $\square$

## 4.2 Proof of the “only if” Part of Theorem 19

From now on,  $S$  will stand for a simple  $(k, \bar{k})$ -surface which in particular, is a  $\bar{k}$ -thin set. Therefore, by Th. 18, it will suffice to prove that its continuous analogue  $|\mathcal{A}_S|$  is a combinatorial surface; that is,  $\text{lk}(c(\delta), \mathcal{A}_S)$  is a 1-sphere for each cell  $\delta \in R^3$  such that  $f_{k\bar{k}}(S, \delta) = 1$ . For this notice that  $\text{lk}(c(\delta), \mathcal{A}_S)$  is the full subcomplex of  $\mathcal{A}_{R^3}$  determined by the centres  $c(\gamma)$  of all cells  $\gamma < \delta$  or  $\gamma > \delta$  with  $f_{k\bar{k}}(S, \gamma) = 1$ . The proof will consist in an analysis of cases according to the dimension of the cell  $\delta$ . This analysis benefits from the work already done in Prop. 9 and uses the sets of patterns  $\mathbb{P}_{\bar{k}}$  involved in the definition of the function  $f_{k\bar{k}}$  (Def. 12). Recall that if  $K \cap S \in \mathbb{P}_{\bar{k}}$  for a given unit cube  $K \subseteq \mathbb{Z}^3$  then  $K - S$  is not  $\bar{k}$ -connected and, by the definition of simple  $(k, \bar{k})$ -surface, for each  $\sigma \in K \cap S$ , it contains voxels of the  $\bar{k}$ -components,  $A_\sigma, B_\sigma$ , of  $\bar{S}_\sigma$  which are  $\bar{k}$ -adjacent to  $\sigma$ ; see Remark 13 and Def. 8.

**Proposition 22.** *Let  $\beta \in R^3$  be a 1-cell with  $f_{k\bar{k}}(S, \beta) = 1$ , and let  $K_\beta$  be the unit square centred at  $\beta$ . Then, one of the two following properties holds, and in both cases  $\text{lk}(c(\beta), \mathcal{A}_S)$  is a 1-sphere.*

1.  $K_\beta \subseteq S$  and  $f_{k\bar{k}}(S, \alpha_1) = 0 = f_{k\bar{k}}(S, \alpha_2)$  for the two vertices  $\alpha_1, \alpha_2 < \beta$ .
2.  $K_\beta \cap S = \{\sigma, \tau\}$ ,  $\bar{k} = 6$  and  $f_{k\bar{k}}(S, \alpha_1) = 1 = f_{k\bar{k}}(S, \alpha_2)$ .

*Proof.* It is immediate to check that if  $K_\beta \subseteq S$  then  $K_{\alpha_i} \cap S \notin \mathbb{P}_{\bar{k}}$ , where  $K_{\alpha_i}$  is the unit cube of  $\mathbb{Z}^3$  centred at  $\alpha_i$ ,  $i = 1, 2$ . Then, property (1) follows from Prop. 9 and Def. 12. For (2) we know, by Lemma 3, that the  $18^S$ -adjacent voxels  $\sigma, \tau$  are not  $\bar{k}_6$ -linked if  $\bar{k} = 18, 26$  and, moreover, any  $\bar{k}_6$ -path between them is actually a 6-path. Then, if  $f_{\bar{k}\bar{k}}(S, \beta) = 1$  and  $K_\beta \not\subseteq S$  necessarily  $K_\beta \cap S = \{\sigma, \tau\}$ ,  $\bar{k} = 6$  and  $f_{\bar{k}\bar{k}}(S, \alpha_1) = f_{\bar{k}\bar{k}}(S, \alpha_2)$  by Def. 12. If  $f_{\bar{k}\bar{k}}(S, \alpha_i) = 0$  then  $K_{\alpha_i} \cap S \notin \mathbb{P}_6$  and, besides  $\sigma$  and  $\tau$ , it contains at most a third voxel which is 6-adjacent to one of them by Prop. 9. Therefore,  $K_{\alpha_i} - S$  is 6-connected by Remark 13 and it is easily checked that it does not contain a  $\bar{k}_6$ -path from  $\sigma$  to  $\tau$ . Hence,  $\sigma, \tau$  are not  $\bar{k}_6$ -linked, since  $\overline{O}_\sigma \cap \overline{O}_\tau = (K_{\alpha_i} \cup K_{\alpha_j}) \cap S$  is 6-connected, nor  $\bar{k}_{18}$ -linked.  $\square$

**Proposition 23.** *Let  $\alpha \in R^3$  be a 0-cell with  $f_{\bar{k}\bar{k}}(S, \alpha) = 1$ , and let  $K_\alpha$  be the unit square centred at  $\alpha$ . Then, one of the following properties holds, and in all these cases  $\text{lk}(c(\alpha), \mathcal{A}_S)$  is readily checked to be a 1-sphere.*

1.  $\bar{k} \in \{6, 18\}$  and  $K_\alpha \cap S$  corresponds to pattern  $\mathbb{P}_6^c$ .
2.  $\bar{k} = 6$  and  $K_\alpha \cap S$  corresponds to a pattern in the set  $\mathbb{P}_6 - \{\mathbb{P}_4^f, \mathbb{P}_6^c\}$ ; then  $f_{k6}(S, \sigma \cap \tau) = 1$  for each pair of  $18^S$ -adjacent voxels  $\sigma, \tau \in K_\alpha \cap S$ .
3.  $\bar{k} = 6$  and  $K_\alpha \cap S = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}$  corresponds to the pattern  $\mathbb{P}_4^f$ ; then  $f_{k6}(S, \sigma_i \cap \sigma_{i+1 \bmod 4}) = 1$ ,  $0 \leq i \leq 3$ , while  $f_{k6}(S, \sigma_j \cap \sigma_{j+2}) = 0$  for  $j = 0, 1$ .

*Proof.* By Prop. 9 and Def. 12 we know that  $K_\alpha \cap S \in \mathbb{P}_{\bar{k}}$  and hence  $\bar{k} = 6$  if  $K_\alpha \cap S \neq \mathbb{P}_6^c$ . Given  $\sigma, \tau \in K_\alpha \cap S$  two  $18^S$ -adjacent voxels, let  $\alpha, \alpha'$  the vertices of the 1-cell  $\sigma \cap \tau$ . Notice that  $K_\alpha - S$  has exactly two 6-components whenever  $K_\alpha \cap S \in \mathbb{P}_6 - \{\mathbb{P}_4^f\}$ , one contained in  $A_\sigma$  and the other in  $B_\sigma$  since  $S$  is simple  $(k, 6)$ -surface. Therefore  $K_{\alpha'} - S$  is not 6-connected too. Hence  $K_{\alpha'} \cap S \in \mathbb{P}_6$  and (2) follows by the definition of  $f_{k6}$ .

For the proof of (3) we have some choices to make. Let  $\rho_i \in K_\alpha - S$  be the voxel strictly 26-adjacent to  $\sigma_i$  and notice that  $\rho_i \notin A_{\sigma_i} \cup B_{\sigma_i}$ , while  $A_{\sigma_i}$  and  $B_{\sigma_i}$  contain at least one of the other three voxels in  $K_\alpha - S$ , which are 6-adjacent to  $\sigma_i$ . Without loss of generality assume that  $\rho_1, \rho_3 \in A_{\sigma_0}$  and  $\rho_2 \in B_{\sigma_0}$ . The argument used in the previous case also shows that  $f_{k6}(S, \sigma_0 \cap \sigma_i) = 1$  for  $i = 1, 3$ . To check that  $f_{k6}(S, \sigma_0 \cap \sigma_2) = 0$  it suffices to prove that  $K_{\alpha'} - S$  is 6-connected, where  $\alpha' \neq \alpha$  is the other vertex of the 1-cell  $\sigma_0 \cap \sigma_2$ . Otherwise, if  $K_{\alpha'} - S$  is not 6-connected, the voxels in  $K_{\alpha'} - K_\alpha$  which are 6-adjacent to  $\rho_1$  and  $\rho_2$  are necessarily in  $S$ , while the voxel  $\tau_0 \in K_{\alpha'} - K_\alpha$  6-adjacent to  $\sigma_0$  is in  $B_\sigma$ . However, under these conditions it is not difficult to check that any 6-path in  $\overline{O}_{\sigma_0}$  from  $\rho_2$  to  $\tau_0$  crosses any 6-path from  $\rho_1$  to  $\rho_3$ . Now, the same argument applied to  $\sigma_2$  and  $\sigma_1$  yields that  $f_{k6}(S, \sigma_2 \cap \sigma_i) = 1$ , for  $i = 1, 3$  (since  $f_{k6}(S, \sigma_0 \cap \sigma_2) = 0$ ) and  $f_{k6}(S, \sigma_1 \cap \sigma_3) = 0$ , respectively.  $\square$

**Lemma 24.** *Let  $\beta \in R^3$  be a 1-cell with vertices  $\alpha_1, \alpha_2$ . If  $K_\beta \subseteq S$  then  $K_{\alpha_1} - S$  and  $K_{\alpha_2} - S$  are  $\bar{k}$ -connected and, for any voxel  $\sigma \in K_\beta$ , these differences are contained in distinct  $\bar{k}$ -components of  $\overline{O}_\sigma$   $\bar{k}$ -adjacent to  $\sigma$ .*

**Proposition 25.** *Let  $\gamma \in R^3$  be a 2-cell with  $f_{\bar{k}\bar{k}}(S, \alpha) = 1$ ; that is,  $\gamma$  is the intersection of two 6-adjacent voxels,  $\sigma, \tau \in S$ . Then  $f_{\bar{k}\bar{k}}(S, \delta_i) = 1$  for exactly two faces  $\delta_1, \delta_2 < \gamma$  and, moreover,  $\delta_1 \not\prec \delta_2$ .*

*Proof.* Notice that  $D_\gamma = |\text{st}(c(\gamma), \mathcal{A}_{R^3})| - |\text{st}(c(\gamma), \mathcal{A}_S)| \subseteq D_\sigma \cap D_\tau$  since any face of  $\gamma$  is also a face of  $\sigma$  and  $\tau$ . Then, if  $D_\gamma$  is connected  $\cup_{c(\delta) \in D_\gamma} K_\delta - S = \overline{O}_\sigma \cap \overline{O}_\tau$  is contained in a  $\overline{k}$ -component of both  $\overline{O}_\sigma$  and  $\overline{O}_\tau$ , which is a contradiction with Lemma 3. Therefore,  $f_{\overline{k}\overline{k}}(S, \delta) = 1$  for, at least, two faces of  $\gamma$ .

Let  $\alpha_i$  and  $\beta_i$  be the vertices and edges of  $\gamma$ , respectively, with  $\alpha_i, \alpha_{i+1 \bmod 4} < \beta_i, 0 \leq i \leq 3$ . If  $f_{\overline{k}\overline{k}}(S, \beta_i) = 1$  then the unit square  $K_{\beta_i}$  is contained in  $S$  and we know, by Prop. 22, that  $f_{\overline{k}\overline{k}}(S, \alpha_i) = 0 = f_{\overline{k}\overline{k}}(S, \alpha_{i+1})$  and  $K_{\alpha_i} - S \subseteq A_\sigma, K_{\alpha_{i+1}} - S \subseteq B_\sigma$  by Lemma 24. On the other hand, if  $f_{\overline{k}\overline{k}}(S, \alpha_i) = 1$ , then  $\overline{k} = 6, 18$  and  $f_{\overline{k}\overline{k}}(S, \beta_i) = 0 = f_{\overline{k}\overline{k}}(S, \beta_{i-1})$ . Moreover, it is readily checked that  $K_{\beta_i} - S$  and  $K_{\beta_{i-1}} - S$  are in distinct 6-components of  $K_\gamma - S$  and thus one is contained in  $A_\sigma$  while  $B_\sigma$  contains the other. Therefore, if  $f_{\overline{k}\overline{k}}$  lights more than two faces of  $\gamma$  then it lights exactly either all its vertices or all its edges. Moreover, in the second case  $K_{\alpha_0} \cup K_{\alpha_2} - S \subseteq A_\sigma$  while  $K_{\alpha_1} \cup K_{\alpha_3} - S \subseteq B_\sigma$ . However, ones easily check that any  $\overline{k}$ -path from  $K_{\alpha_0} - S$  to  $K_{\alpha_2} - S$  must be contained in  $\overline{O}_\sigma - N_{26}(\tau)$  and then it crosses any  $\overline{k}$ -path from  $K_{\alpha_1} - S$  to  $K_{\alpha_3} - S$ , which is a contradiction.  $\square$

**Lemma 26.** *Let  $\beta \in R^3$  be a 1-cell which is a face of some voxel  $\sigma \in S$ . The two following properties hold for the subset of simplices  $X_\beta \subseteq \text{lk}(c(\sigma), \mathcal{A}_S)$  whose vertices are in  $\{c(\delta); K_\delta \subseteq K_{\alpha_1} \cup K_{\alpha_2}\}$ , where  $\alpha_1, \alpha_2$  are the two vertices of  $\beta$ .*

1. *If  $\overline{k} = 6$  then  $X_\beta$  is not a 1-sphere.*
2. *If  $\overline{k} \in \{18, 26\}$  and  $X_\beta$  is a 1-sphere then  $\cup_\alpha K_\alpha - S$  is contained in one of the  $\overline{k}$ -components of  $\overline{O}_\sigma$  which are  $\overline{k}$ -adjacent to  $\sigma$ , where  $\alpha$  ranges over the vertices of  $\sigma$  distinct from  $\alpha_1$  and  $\alpha_2$ .*

*Proof (Sketch).* There are exactly three configurations in  $(K_{\alpha_1} \cup K_{\alpha_2}) \cap S$  for which  $X_\beta$  is a 1-sphere:

- (a) For some  $i = 1, 2$ ,  $K_{\alpha_i} \cap S = P_7$  consists of three unit squares containing  $\sigma$ .
- (b)  $(K_{\alpha_1} \cup K_{\alpha_2}) \cap S$  consists of four unit squares, distinct from  $K_\beta$ , containing  $\sigma$ .
- (c)  $K_{\alpha_1} \cap S \in \mathbb{P}_{\overline{k}}^f$  while  $K_{\alpha_2} \cap S$  is either in  $\mathbb{P}_{\overline{k}}^f$  or it consists of two unit squares.

From the definition of  $f_{\overline{k}\overline{k}}$  we know that case (c) is not possible for  $\overline{k} = 26$  while, by Prop. 9, case (a) is only possible for this adjacency. Then  $K_{\alpha_i} - S = \{\mu\}$  is a 26-component of  $\overline{O}_\sigma$  and property (2) follows in this case. On the other hand, in cases (b) and (c) the voxel  $\tau \in K_\beta$  strictly 18-adjacent to  $\sigma$  is not in  $S$  and it can be checked that it belongs to a  $\overline{k}$ -component  $C$  of  $\overline{O}_\sigma$  which is contained in  $(K_{\alpha_1} \cup K_{\alpha_2}) - S$ . Therefore,  $C \in \{A_\sigma, B_\sigma\}$  if  $\overline{k} \in \{18, 26\}$  and the proof of (2) is completed. However,  $C$  is not 6-adjacent to  $\sigma$ . Therefore, case (c) is not possible for  $\overline{k} = 6$  since  $\tau \in K_{\alpha_1} \cap S$  and this set corresponds to a pattern in  $\mathbb{P}_6 - \{P_4^f\}$ ; hence  $K_{\alpha_1} - S$  has exactly two 6-components which are contained in  $A_\sigma$  and  $B_\sigma$  by Def. 8.

To complete the proof of (1) it suffices to find a 2-cell  $\gamma < \sigma$  such that  $c(\gamma)$  belongs to three segments in  $\text{lk}(c(\sigma), \mathcal{A}_S)$ , which is a contradiction with Prop. 25. This is readily checked from the fact that  $A_\sigma$  and  $B_\sigma$  share the two only voxels in  $N_{26}(\sigma) - (K_{\alpha_1} \cup K_{\alpha_2})$  which are 6-adjacent to  $\sigma$ .  $\square$

**Proposition 27.** *The link  $\text{lk}(c(\sigma), \mathcal{A}_S)$  is a 1-sphere for each voxel  $\sigma \in S$ .*

*Proof (Sketch).* It is not hard to derive from Prop. 22, 23 and 25 that  $\text{lk}(c(\sigma), \mathcal{A}_S)$  is the disjoint union of a non-empty set of 1-spheres. Next we check that this set consists of just one 1-sphere.

Firstly, assume that there exists a voxel  $\tau \in S$  6-adjacent to  $\sigma$ , and let  $L_\gamma \subseteq \text{lk}(c(\sigma), \mathcal{A}_S)$  be the 1-sphere that contains the centre  $c(\gamma)$  of the 2-cell  $\gamma = \sigma \cap \tau$ . In addition, let  $\delta < \gamma$  be one of the two cells with  $f_{\overline{k\bar{k}}}(S, \delta) = 1$  provided by Prop. 25. Then  $c(\gamma), c(\delta), c(\sigma \cap \rho) \in L_\gamma$  for any other voxel  $\rho \in S$  6-adjacent to  $\sigma$  and such that  $\delta < \rho$ . Moreover,  $c(\delta') \in L_\gamma$  for any face of  $\gamma$  or  $\sigma \cap \rho$  with  $f_{\overline{k\bar{k}}}(S, \delta') = 1$ . Hence, the vertices of some other 1-sphere in  $\text{lk}(c(\sigma), \mathcal{A}_S)$  should be centres of cells which are the intersection of voxels in  $K_{\alpha_1} \cup K_{\alpha_2}$ , where  $\alpha_1, \alpha_2$  are the two vertices of  $\sigma$  which are not vertices of  $\tau$  or  $\rho$ . However, notice that if  $\delta$  is a 0-cell then  $K_\delta - S$  is not  $\overline{k}$ -connected, by Remark 13, and it contains voxels of  $A_\sigma$  and  $B_\sigma$  by Def. 8. Otherwise, if  $\delta$  is a 1-cell then it is the centre of the unit square  $K_\delta$  and, according to Lemma 24,  $K_{\alpha_3} - S \subseteq A_\sigma$ ,  $K_{\alpha_4} - S \subseteq B_\sigma$ , where  $\alpha_3, \alpha_4$  are the vertices of  $\delta$ . Therefore, Lemma 26 yields that no other 1-sphere in  $\text{lk}(c(\sigma), \mathcal{A}_S)$  can be determined by the voxels in  $K_{\alpha_1} \cup K_{\alpha_2}$ .

Using a similar but more elaborate argument we reach the same conclusion if the voxel  $\rho > \delta$  is strictly 18-adjacent to  $\sigma$ . Thus, we are done if we prove that  $\text{lk}(c(\sigma), \mathcal{A}_S)$  consists of a single 1-sphere in case the six voxels  $\{\tau_i\}_{i=1}^6 = N_6(\sigma) - \{\sigma\}$  are not in  $S$  and, thus, they are shared by  $A_\sigma$  and  $B_\sigma$ . This follows by analysing the four possible configurations: (1)  $A_\sigma$  contains just the voxel  $\tau_1$ ; (2)  $\{\tau_1, \tau_2\} \subseteq A_\sigma$  where, in addition,  $\tau_1$  and  $\tau_2$  are proved to be 18-adjacent; (3)  $\{\tau_i\}_{i=1}^3 \subseteq A_\sigma$  (two configurations). All these configurations are easily worked out by taking into account that if  $\tau_i \in A_\sigma$  and  $\tau_j \in B_\sigma$  then any voxel which is 6-adjacent to both lies in  $S$  since this case can occur only if  $\overline{k} = 6$ .  $\square$

## References

1. Ayala, R., Domínguez, E., Francés, A.R., Quintero, A.: Weak lighting functions and strong 26-surfaces. *Theoretical Computer Science* 283(1), 29–66 (2002)
2. Bertrand, G., Malgouyres, R.: Some topological properties of surfaces in  $\mathbb{Z}^3$ . *Journal of Mathematical Imaging and Vision* 11, 207–221 (1999)
3. Ciria, J.C., Domínguez, E., Francés, A.R., Quintero, A.: A plate-based definition of discrete surfaces. *Pattern Recogn. Lett.* 33(11), 1485–1494 (2012)
4. Ciria, J.C., Domínguez, E., Francés, A.R.: Separation Theorems for Simplicity 26-Surfaces. In: Braquelaire, A., Lachaud, J.-O., Vialard, A. (eds.) *DGCI 2002*. LNCS, vol. 2301, pp. 45–56. Springer, Heidelberg (2002)
5. Ciria, J.C., Domínguez, E., Francés, A.R., Quintero, A.: Universal Spaces for  $(k, \overline{k})$ -Surfaces. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 385–396. Springer, Heidelberg (2009)
6. Couprie, M., Bertrand, G.: Simplicity surfaces: a new definition of surfaces in  $\mathbb{Z}^3$ . In: *SPIE Vision Geometry V*, vol. 3454, pp. 40–51 (1998)
7. Kong, T.Y., Roscoe, A.W.: Continuous analogs of axiomatized digital surfaces. *Computer Vision, Graphics, and Image Processing* 29(1), 60–86 (1985)
8. Morgenthaler, D.G., Rosenfeld, A.: Surfaces in three-dimensional digital images. *Information and Control* 51(3), 227–247 (1981)

# A Parallel Thinning Algorithm for Grayscale Images

Michel Couprie<sup>1</sup>, Nivando Bezerra<sup>2</sup>, and Gilles Bertrand<sup>1,\*</sup>

<sup>1</sup> Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge, Équipe A3SI,  
ESIEE Paris, France

<sup>2</sup> Instituto Federal do Ceará, IFCE, Maracanaú, Brazil

**Abstract.** Grayscale skeletonization offers an interesting alternative to traditional skeletonization following a binarization. It is well known that parallel algorithms for skeletonization outperform sequential ones in terms of quality of results, yet no general and well defined framework has been proposed until now for parallel grayscale thinning. We introduce in this paper a parallel thinning algorithm for grayscale images, and prove its topological soundness based on properties of the critical kernels framework. The algorithm and its proof, given here in the 2D case, are also valid in 3D. Some applications are sketched in conclusion.

## 1 Introduction

Topology-preserving transformations, in particular topology-preserving thinning and skeletonization, are essential tools in many applications of image processing. In the huge literature dealing with this topic, almost all works are devoted to the case of binary images. Even so, there are cases when thinning directly a grayscale image, instead of a binarization of this one, can be beneficial [23, 1, 9, 13]. First, binarization unusually involves important information loss, and it may be desirable to defer this loss to the latest steps of the processing chain. Second, working with full grayscale information permits to detect and to use specific features, such as crests and valleys, peaks and wells, or saddle points. These features can be precisely defined within the framework exposed in this paper.

Some attention has been given to the development of thinning algorithms acting directly on grayscale images. Dyer and Rosenfeld [11] proposed an algorithm based on a notion of weighted connectedness. The thinning is done directly over the graylevel values of the points but, as pointed out in the same paper [11], the connectivity of objects is not always preserved. Thinning based on a fuzzy framework for image processing has been proposed in [22, 20], but also in this case object connectedness is not ensured in the final skeleton. The more recent works in [25, 2] use an implicit image binarization into a background and a grayscale foreground.

---

\* This work has been partially supported by the “ANR-2010-BLAN-0205 KIDICO” project.

Other approaches for grayscale thinning (that is, thinning of grayscale images without prior segmentation, resulting in either a grayscale or a binary skeleton) are pseudo distance maps [19, 12], pixel superiority index [13], and partial differential equations (see *e.g.* [16]). In all these works, no property relative to topology preservation is claimed.

In this paper, we adopt a topological approach, beginning with a definition of the topological equivalence between two maps. This definition is based on the decomposition of a map into its different sections [8, 7]: let  $F$  be a map from  $\mathbb{Z}^2$  into  $\mathbb{Z}$ , the section of  $F$  at level  $k$  is the set  $F_k$  of points  $x$  in  $\mathbb{Z}^2$  such that  $F(x) \geq k$ . Following this approach called *cross-section topology*, a transformation is homotopic, i.e. preserves the topology of  $F$ , if it preserves the topology in the binary sense of every section  $F_k$ . An elementary homotopic transformation consists of lowering the value of a so-called destructible point (a notion introduced in [6], which generalizes the notion of simple point [15] to maps). Based on this elementary operation, sequential thinning algorithms for grayscale images have been proposed in [7, 10], with applications to image segmentation, filtering and restoration.

By nature, all these sequential thinning algorithms have the drawback of producing a result that depends on arbitrary choices that must be done, with regard to the order in which the destructible points are treated.

On the other hand, although parallel thinning of binary images is a quite well developed topic in the image processing community, with thousands of references, very few attempts have been made until now to propose parallel grayscale thinning algorithms. In [18], an algorithm was proposed but no well-stated property and no proof of topological correctness was given. The first (to our best knowledge) approach for parallel grayscale thinning with proved properties was introduced by [17], in the framework of partial orders. Here, the result is a map which is defined on a space which is not the classical pixel grid, but can be seen as a grid with higher resolution. Finally, [21] introduces order-independent thinning for both binary and grayscale images. However their definition is combinatorial in nature, and does not lead to efficient algorithms.

The approach taken in this paper is based on the framework of critical kernels [3], which is to our knowledge the most general framework to analyze and design parallel homotopic thinning algorithms in discrete spaces, with the guarantee of topology preservation. Our main contribution are algorithm 1, which simultaneously considers all pixels of a grayscale image and lowers some of them in one thinning step, and the proof of its topological soundness (theorem 14). We conclude the paper by an illustration of the algorithm and some applications.

## 2 Parallel Topological Transformations of Binary Images

As we base our notion of topological equivalence for functions on the one for sets (or binary images), we begin by providing some definitions and results for this latter case. The framework of critical kernels, introduced by one of the authors in [3], will serve us to prove the topological soundness of the proposed method.



This framework is established within the context of simplicial or cubical complexes, however the resulting algorithms can be directly implemented in  $\mathbb{Z}^2$  thanks to very simple masks. Only a small set of definitions and properties based on cubical complexes are needed to understand the rest of the paper.

Intuitively, a cubical complex may be thought of as a set of elements having various dimensions (*e.g.* squares, edges, vertices) glued together according to certain rules.

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  $\mathbb{Z}^2$ , which is the Cartesian product of exactly  $d$  elements of  $\mathbb{F}_1^1$  and  $(2 - d)$  elements of  $\mathbb{F}_0^1$  is called a *face* or a  $d$ -*face* in  $\mathbb{Z}^2$ ,  $d$  is the *dimension* of  $f$ , we write  $\dim(f) = d$ .

A  $d$ -face is called a *point* if  $d = 0$ , a (*unit*) *edge* if  $d = 1$ , a (*unit*) *square* or a *pixel* if  $d = 2$ . We denote by  $\mathbb{P}^2$  the set composed of all 2-faces (pixels) in  $\mathbb{Z}^2$ . We denote by  $\mathcal{P}$  the collection of all finite sets which are composed solely of pixels.

Let  $x, y$  be two pixels, let  $d \in \{0, 1\}$ . We say that  $x$  and  $y$  are  $d$ -*adjacent* if there is  $k, 2 \geq k \geq d$ , such that  $x \cap y$  is a  $k$ -face. We write  $\mathcal{N}_d(x)$  to denote the set of all pixels that are  $d$ -adjacent to  $x$ . Note that for any pixel  $x$  and any  $d$ , we have  $x \in \mathcal{N}_d(x)$ . We set  $\mathcal{N}_d^*(x) = \mathcal{N}_d(x) \setminus x$ . Remark that we have 4 (resp. 8) pixels in  $\mathcal{N}_1^*(x)$  (resp.  $\mathcal{N}_0^*(x)$ ). Let  $Y$  be a set of pixels, we say that  $x$  and  $Y$  are  $d$ -*adjacent* if there exists a pixel  $y$  in  $Y$  such that  $x$  and  $y$  are  $d$ -adjacent.

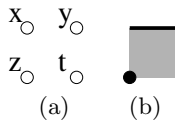
Let  $X \in \mathcal{P}$  and let  $Y \subseteq X, Y \neq \emptyset$ . We say that  $Y$  is  $d$ -*connected* in  $X$  if, for any  $x, y \in Y$ , there exists a sequence  $\langle x_0, \dots, x_\ell \rangle$  of pixels of  $X$ , such that  $x_0 = x, x_\ell = y$ , and for any  $i \in \{1, \dots, \ell\}$ ,  $x_i$  is  $d$ -adjacent to  $x_{i-1}$ . We say that  $Y$  is a  $d$ -*connected component* of  $X$  if  $Y$  is  $d$ -connected in  $X$  and if it is maximal for the inclusion, that is, we have  $Y = Z$  whenever  $Y \subseteq Z \subseteq X$  and  $Z$  is  $d$ -connected in  $X$ .

Let  $X \in \mathcal{P}$  and let  $x \in X$ . We denote by  $\overline{X}$  the complementary set of  $X$ , that is,  $\overline{X} = \mathbb{P}^2 \setminus X$ . We denote by  $T(x, X)$  the number of 0-connected components of  $\mathcal{N}_0^*(x) \cap X$ . We denote by  $\overline{T}(x, X)$  the number of 1-connected components of  $\mathcal{N}_0^*(x) \cap \overline{X}$  that are 1-adjacent to  $x$ .

Intuitively, a pixel  $x$  in a set  $X$  of pixels is simple if its removal from  $X$  “does not change the topology of  $X$ ”. We recall here a definition of a simple pixel, which is based on the following recursive definition.

**Definition 1** ([5]). Let  $X \in \mathcal{P}$ . We say that  $X$  is a *reducible set* if either:

- i)  $X$  is composed of a single pixel, or
- ii) there exists  $x \in X$  such that  $\mathcal{N}_0^*(x) \cap X$  is a reducible set and  $X \setminus x$  is a reducible set.



**Fig. 1.** (a): Four elements  $x, y, z, t$  of  $\mathbb{Z}^2$ . (b): A graphical representation of the set of faces  $\{\{x, y, z, t\}, \{x, y\}, \{z\}\}$ : a pixel, an edge, and a point.

**Definition 2** ([5]). Let  $X \in \mathcal{P}$ . A pixel  $x \in X$  is *simple for  $X$*  if  $\mathcal{N}_0^*(x) \cap X$  is a reducible set. If  $x$  is simple for  $X$ , we say that  $X \setminus x$  is an *elementary thinning* of  $X$ .

Let  $X, Y \in \mathcal{P}$ . We say that  $Y$  is a *thinning of  $X$*  if there exists a sequence  $\langle X_0, \dots, X_\ell \rangle$  such that  $X_0 = X$ ,  $X_\ell = Y$ , and for any  $i \in \{1, \dots, \ell\}$ ,  $X_i$  is an elementary thinning of  $X_{i-1}$ .

In [5] it has been shown that the above definition of a simple pixel is equivalent to a definition based on the notion of collapse [24], this operation being a discrete analogue of a continuous deformation (a homotopy). Furthermore, the following proposition, which is a straightforward consequence of Prop. 8 [5], shows that definition 2 leads to a characterization of simple pixels which is equivalent to previously proposed ones (see *e.g.* [14]).

**Proposition 3.** Let  $X \in \mathcal{P}$  and let  $x \in X$ . The pixel  $x$  is simple for  $X$  if and only if  $T(x, X) = \overline{T}(x, X) = 1$ .

Now, we are ready to give a short introduction to the framework of critical kernels [3], which is to our knowledge the most powerful framework to study and design parallel topology-preserving algorithms in discrete spaces. We limit ourselves to a minimal yet sufficient set of notions, interested readers may refer to [3, 4, 5] for a more complete presentation.

Let  $C \in \mathcal{P}$ , let  $d \in \{0, 1, 2\}$ . We say that  $C$  is a  *$d$ -clique*, or simply a *clique*, if  $\cap\{x \in C\}$ , the intersection of all pixels in  $C$ , is a  *$d$ -face*.

Let  $X \in \mathcal{P}$  and let  $C \subseteq X$  be a clique. We say that  $C$  is *essential for  $X$*  if we have  $D = C$  whenever  $D$  is a clique such that:

- i)  $C \subseteq D \subseteq X$ , and
- ii)  $\cap\{x \in D\} = \cap\{x \in C\}$ .

Remark that, if  $C$  is composed of a single pixel (*i.e.*  $C$  is a 2-clique), then  $C$  is necessarily essential.

**Definition 4** ([5]). Let  $S \in \mathcal{P}$ . The  *$\mathcal{K}$ -neighborhood* of  $S$ , written  $\mathcal{K}(S)$ , is the set made of all pixels that are 0-adjacent to each pixel in  $S$ . We set  $\mathcal{K}^*(S) = \mathcal{K}(S) \setminus S$ .

Notice that we have  $\mathcal{K}(S) = \mathcal{N}_0(x)$  if and only if  $S$  is made of a single pixel  $x$ .

**Definition 5** ([5]). Let  $X \in \mathcal{P}$  and let  $C$  be a clique that is essential for  $X$ . We say that the clique  $C$  is *regular for  $X$*  if  $\mathcal{K}^*(C) \cap X$  is a reducible set. We say that  $C$  is *critical for  $X$*  whenever  $C$  is not regular for  $X$ .

Remark that, if  $C$  is a singleton  $\{x\}$ , the clique  $C$  is regular whenever  $x$  is simple.

The following result is a consequence of a general theorem which holds for complexes of arbitrary dimension (see [3], Th. 4.2).

**Theorem 6** ([5]). Let  $X \in \mathcal{P}$  and let  $Y \subseteq X$ . If any clique that is critical for  $X$  contains at least one pixel of  $Y$ , then  $Y$  is a thinning of  $X$ .

Our goal is to define a subset of an object  $X$  that contains at least one pixel of each critical clique. We also want this subset to be as small as possible, in order

to obtain an efficient thinning procedure. This motivates the following definition, where the set  $K$  plays the role of a constraint set (that is, a set of pixels that must be preserved from deletion, for other reasons than topology preservation).

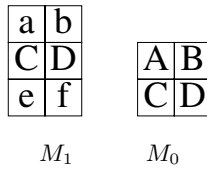
**Definition 7** ([5]). Let  $X \in \mathcal{P}$ , let  $K \in \mathcal{P}$ , and let  $C \subseteq X \setminus K$  be a  $d$ -clique that is critical for  $X$ ,  $d \in \{0, 1, 2\}$ . We say that the clique  $C$  is  $d$ -crucial (or crucial) for  $\langle X, K \rangle$  if

- i)  $d = 2$ , or
- ii)  $d = 1$  and  $C$  does not contain any non-simple pixel, or
- iii)  $d = 0$  and  $C$  does not contain any non-simple pixel, nor any pixel belonging to a 1-clique which is crucial for  $\langle X, K \rangle$ .

The following corollary directly follows from theorem 6.

**Corollary 8** ([5]). Let  $X \in \mathcal{P}$  and let  $Y \subseteq X$ . If any clique that is crucial for  $X$  contains at least one pixel of  $Y$ , then  $Y$  is a thinning of  $X$ .

The following proposition allows us to characterize crucial cliques by the use of only two masks, which apply directly to any object represented by a set of pixels (there is no need to consider the underlying cubical complex, nor to check the condition of definition 5).



**Fig. 2.** Masks for 1-crucial ( $M_1$ ) and 0-crucial ( $M_0$ ) pixels

The masks  $M_1, M_0$  are given in figure 2. For the mask  $M_1$ , we also consider the mask obtained from it by applying a  $\pi/2$  rotation: we get 3 masks (2 for  $M_1$ , and 1 for  $M_0$ ).

**Definition 9.** Let  $X \in \mathcal{P}$ , and let  $M$  be a set of pixels of  $X$ .

- 1) The set  $M$  matches the mask  $M_1$  if:
  - i)  $M = \{C, D\}$ ; and
  - ii) the pixels  $C, D$  are simple for  $X$ ; and
  - iii) the sets  $\{a, b\} \cap X$  and  $\{e, f\} \cap X$  are either both empty or both non-empty.
- 2) The set  $M$  matches the mask  $M_0$  if:
  - i)  $M = \{A, B, C, D\} \cap X$ ; and
  - ii) the pixels in  $M$  are simple and not matched by  $M_1$ ; and
  - iii) at least one of the sets  $\{A, D\}, \{B, C\}$  is a subset of  $M$ .

**Proposition 10.** Let  $X \in \mathcal{P}$ ,  $K \subseteq X$ , and let  $M$  be a set of pixels in  $X \setminus K$  that are simple for  $X$ .

Then,  $M$  is a crucial clique for  $\langle X, K \rangle$  if and only if  $M$  matches the mask  $M_0$  or the mask  $M_1$ .

This proposition was proved with the help of a computer program, by examination of all possible configurations (see also [5] for similar characterizations in 3D).

### 3 Parallel Thinning for Grayscale Images

In this section, topological notions such as those of simple pixel, thinning, crucial clique, are extended to the case of grayscale images. Then, we introduce our parallel thinning algorithm and prove its topological properties.

A 2D grayscale image can be seen as a function  $F$  from  $\mathbb{P}^2$  into  $\mathbb{Z}$ . For each pixel  $x$  of  $\mathbb{P}^2$ ,  $F(x)$  is the gray level, or the luminosity of  $x$ . The *support* of  $F$  is the set of pixels  $x$  such that  $F(x) > 0$ , denoted by  $\text{Supp}(F)$ . We denote by  $\mathcal{F}$  the set of all functions from  $\mathbb{P}^2$  into  $\mathbb{Z}$  that have a finite support.

Let  $F \in \mathcal{F}$  and  $k \in \mathbb{Z}$ , the *cross-section (or threshold) of  $F$  at level  $k$*  is the set  $F_k$  composed of all pixels  $x \in \mathbb{P}^2$  such that  $F(x) \geq k$ . Observe that a cross-section is a set of pixels, that is, a binary image.

Intuitively, we say that a transformation of  $F$  preserves topology if topology of all cross-sections of  $F$  is preserved. Hence, the “cross-section topology” of a function (*i.e.*, of a grayscale image) directly derives from the topology of binary images [7]. Based on this idea, the following notion generalize the notion of simple pixel to the case of functions.

**Definition 11** ([7]). Let  $F \in \mathcal{F}$ ,  $x \in \mathbb{P}^2$ , and  $k = F(x)$ . The pixel  $x$  is *destructible (for  $F$ )* if  $x$  is simple for  $F_k$ . If  $x$  is destructible for  $F$ , we say that the map  $F'$  defined by:

$$F'(y) = \begin{cases} F(x) - 1 & \text{if } y = x, \\ F(y) & \text{otherwise} \end{cases}$$

is an *elementary thinning* of  $F$ .

Let  $F, G \in \mathcal{F}$ . We say that  $G$  is a *thinning of  $F$*  if there exists a sequence  $\langle F_0, \dots, F_\ell \rangle$  such that  $F_0 = F$ ,  $F_\ell = G$ , and for any  $i \in \{1, \dots, \ell\}$ ,  $F_i$  is an elementary thinning of  $F_{i-1}$ .

Intuitively, the gray level of a destructible pixel may be lowered of one unit, while preserving the topology of  $F$ .

We define also:

$$\begin{aligned} \mathcal{N}^{--}(x) &= \{y \in \mathcal{N}_0^*(x); F(y) < F(x)\} \\ F^-(x) &= \begin{cases} \max\{F(y); y \in \mathcal{N}^{--}(x)\}, & \text{if } \mathcal{N}^{--}(x) \neq \emptyset \\ F(x) & \text{otherwise.} \end{cases} \end{aligned}$$

It is easy to see that lowering a destructible pixel  $x$  down to the value  $F^-(x)$  is a topology-preserving transformation. Informally, it is due to the fact that in all the cross-sections from the value  $F(x)$  down to the value  $F^-(x) + 1$ , the neighborhood of  $x$  is the same. The following proposition shows that a more general property holds for cliques that contain  $x$ . Let  $C$  be a clique and  $k \in \mathbb{Z}$ , we denote by  $\mathcal{K}_k(C)$  the  $\mathcal{K}$ -neighborhood of  $C$  in  $F_k$ . In addition, we set  $\mathcal{K}_k^*(C) = \mathcal{K}_k(C) \setminus C$ .

**Proposition 12.** Let  $F \in \mathcal{F}$ , let  $x \in \text{Supp}(F)$ , let  $\ell = F(x)$ . Let  $C$  be a clique of  $F_\ell$  such that  $x \in C$  (possibly  $C = \{x\}$ ). Let  $k = F^-(x)$ .

For any  $j \in \{k + 1, \dots, \ell\}$ , we have  $\mathcal{K}_j^*(C) = \mathcal{K}_\ell^*(C)$ .

The proof is quite easy and left to the reader as an exercise.

Thinning a grayscale image is a useful operation, with applications to image segmentation, filtering, and restoration [7, 10]. Intuitively, this operation extends the minima of an image while reducing its crests to thin lines. In [10], several sequential algorithms to perform this operation have been proposed and studied. Basically, these algorithms consider one destructible point at a time and lower it. Their common drawback lies in the fact that arbitrary choices have to be made concerning the order in which destructible points are considered. In consequence, notions such as the result of a thinning step can hardly be defined with this approach.

Here, we introduce a new thinning algorithm for grayscale images, that lowers points in parallel. Then, we prove that the result of this thinning, which is uniquely defined, can also be obtained through a process that lowers one destructible point at a time: this guarantees the topological soundness of our algorithm.

The following algorithm constitutes one step of parallel thinning. This operation may be repeated a certain number of times, depending on the application, or until stability if one wants to thin an image as much as possible. Furthermore, we introduce as a parameter of the algorithm, a secondary grayscale image  $K$  that plays the role of a constraint: whatever a point  $x$ , it cannot be lowered below the level  $K(x)$ .

---

**Algorithm 1.** ParGrayThinStep( $F, K$ )

---

**Data** :  $F \in \mathcal{F}, K \in \mathcal{F}$  such that  $K \leq F$   
**2**  $D = \{x \in \text{Supp}(F) \mid x \text{ is destructible for } F \text{ and } F(x) \neq K(x)\};$   
**4**  $R = \{x \in D \mid x \text{ is crucial for } \langle F_k, K_k \rangle, \text{ with } k = F(x)\};$   
**6** **foreach**  $x \in \text{Supp}(F)$  **do**  
**8**   **if**  $x \in D \setminus R$  **then**  $G(x) = \max\{F^-(x), K(x)\};$  **else**  $G(x) = F(x);$   
**10** **return**  $G$

---

The next proposition is an essential step for proving the topological soundness of this algorithm (theorem 14).

**Proposition 13.** Let  $F \in \mathcal{F}$ , let  $K \in \mathcal{F}$  such that  $K \leq F$ .

Let  $G = \text{ParGrayThinStep}(F, K)$ .

For any  $k \in \mathbb{Z}, k > 0$ , if  $C$  is a critical clique of  $F_k$ , then  $G_k$  contains at least one pixel of  $C$ .

Proof: Let  $C$  be a critical clique of  $F_k$ , note that  $C$  may be composed of only one, non-simple pixel. If there exist two pixels  $x$  and  $y$  of  $C$  that are such that  $F(x) > F(y)$ , then  $G(x) \geq F^-(x) \geq F(y)$ . As  $F(y) \geq k$ , we have  $x \in G(k)$ , thus  $G_k$  contains at least one pixel of  $C$ .

Now suppose that, for any  $x, y \in C$ ,  $F(x) = F(y) = \ell$ , thus  $\ell \geq k$  (for  $C$  is a clique of  $F_k$ ), and  $C \subseteq F_\ell$ .

Suppose that  $C \cap K_\ell \neq \emptyset$ . Since  $G(x) = \max\{F^-(x), K(x)\}$  (line 8), we have  $G(x) \geq \ell$ , for any  $x \in C \cap K_\ell$ . We have  $C \cap K_\ell \subseteq G_\ell$  and  $C \cap K_\ell \subseteq G_k$  (since  $G_\ell \subseteq G_k$ , for  $\ell \geq k$ ):  $G_k$  contains at least one pixel of  $C$ .

In the sequel, we suppose that  $C \cap K_\ell = \emptyset$ .

The set  $\mathcal{K}_k^*(C)$  is not reducible, for  $C$  is a critical clique of  $F_k$ . We also remark that  $C$  is necessarily an essential clique for  $F_\ell$ .

1) Suppose that  $\mathcal{K}_\ell^*(C)$  is not reducible. This implies that  $C$  is a critical clique of  $F_\ell$ . By definition of a crucial pixel, there exists at least one pixel  $x$  of  $C$  that is crucial for  $F_\ell$  (and, by consequence, for  $\langle F_\ell, K_\ell \rangle$ ). In this case, we have  $x \in R$  (line 4), hence  $G(x) = F(x)$  (line 8), and we have  $x \in G_\ell$  and  $x \in G_k$ .

2) Suppose that  $\mathcal{K}_\ell^*(C)$  is reducible, thus  $C \subseteq D \setminus R$  (line 4). This implies that  $\mathcal{K}_\ell^*(C) \neq \mathcal{K}_k^*(C)$ , and that there exists  $x \in \mathcal{K}_k^*(C)$ ,  $x \notin \mathcal{K}_\ell^*(C)$ . Thus, we have  $F(x) \geq k$  and  $F(x) < \ell$ . If  $y \in C$ , then  $F^-(y) \geq F(x) \geq k$ . Hence  $G(y) \geq k$ , and  $C \subseteq G_k$ . □

Based on the above property, we can now prove the following theorem, which is the main result of this article. Intuitively, it assesses that algorithm ParGrayThinStep is topology-preserving, in the sense of cross-section topology.

**Theorem 14.** Let  $F \in \mathcal{F}$ , let  $K \in \mathcal{F}$  such that  $K \leq F$ .

Let  $G = \text{ParGrayThinStep}(F, K)$ .

Then,  $G$  is a thinning of  $F$ .

Proof: Let  $M = \max\{F(x) \mid x \in \text{Supp}(F)\}$ ,  $m = \min\{F(x) \mid x \in \text{Supp}(F)\}$ . For any  $k \in \{m, \dots, M\}$ , we define the map  $H^{(k)}$  as follows:

$$\text{For any } x \in \text{Supp}(F), H^{(k)}(x) = \begin{cases} G(x) & \text{if } G(x) \geq k, \\ \min\{F(x), k\} & \text{otherwise.} \end{cases}$$

By construction, we have  $H^{(M)} = F$ ,  $H^{(m)} = G$ , and for any  $k \in \{m, \dots, M\}$ , we have  $H_k^{(k)} = F_k$ .

Let  $C$  be any critical clique of  $H_k^{(k)}$ . By proposition 13,  $G_k$  contains at least one pixel of  $C$ . We can see that  $G \leq H^{(k-1)}$  (indeed  $G \leq H^{(j)}$ , for any  $j$ ), hence  $G_k \subseteq H_k^{(k-1)}$  and  $H_k^{(k-1)}$  contains at least one pixel of  $C$ . Thus by theorem 6,  $H_k^{(k-1)}$  is a thinning of  $H_k^{(k)}$ . In other words, there exists a sequence of elementary (binary) thinnings from  $H_k^{(k)}$  to  $H_k^{(k-1)}$ . By construction, to this sequence corresponds a sequence of elementary (grayscale) thinnings from  $H^{(k)}$  to  $H^{(k-1)}$ . Thus  $H^{(k-1)}$  is a thinning of  $H^{(k)}$  for any  $k \in \{m+1, \dots, M\}$ , hence  $G = H^{(m)}$  is a thinning of  $F = H^{(M)}$ . □

Remark: proposition 13, theorem 14 and their proofs hold whatever the (finite) dimension of the space.

### 4 Illustration and Applications

Figure 3 presents an example of gray level thinning. We have a gray level image with 4 dark minima separated by lighter borders, as well as 3 maxima in (a). After one iteration of symmetric parallel thinning, we see in (b) that the “width of the borders” has been reduced. The image in (c) is obtained after 3 iterations, when stability is achieved. We note that all the 4 minima and the 3 maxima are preserved at their original height. The minimal height of the borders separating the minima is also preserved but these borders are thinner and the minima are larger.

However, the borders and the maxima can be further thinned by a variant of our algorithm, called asymmetric parallel thinning. The three maxima in (c), for example, correspond to crucial cliques and are completely preserved by the symmetric thinning algorithm. The variant consists of lowering, in such a configuration, all the points but one. A precise statement and validation of this algorithm will appear in an extended version of this article.

The result of asymmetric parallel thinning applied to (c) is shown in (d). We see that the borders are now even thinner, and each maximum is now reduced to a peak point.

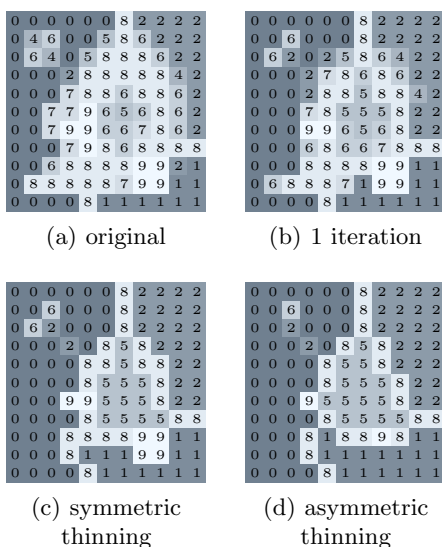
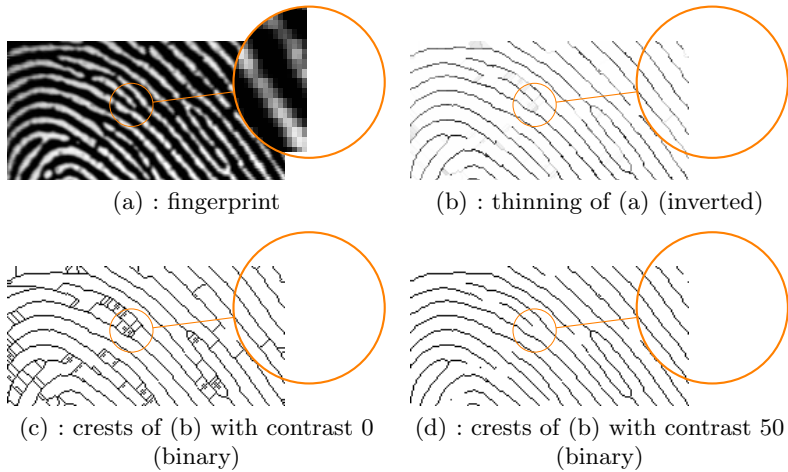
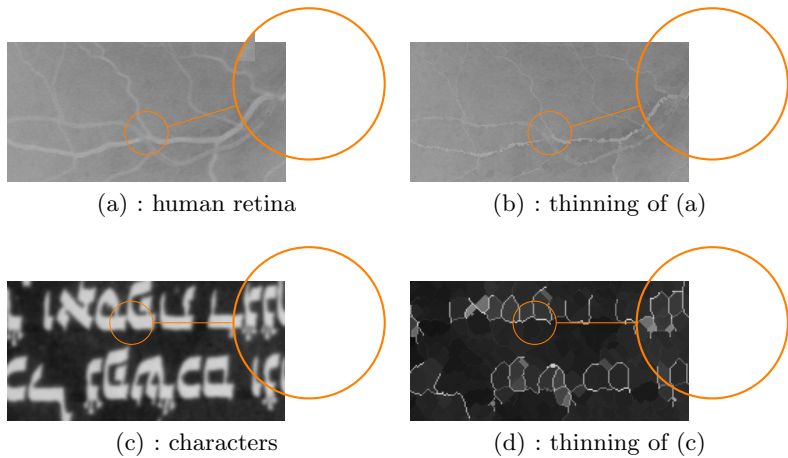


Fig. 3. Gray scale thinning

The gray scale thinning can be used to postpone the binarization process necessary in many applications to obtain a skeleton. This approach allows further processing steps in the richer gray scale space before transforming the image to the more constrained binary image space. In the rest of this section, we show



**Fig. 4.** Fingerprint grayscale thinning and skeleton extraction



**Fig. 5.** Gray scale thinning applications

three examples of applications where grayscale skeletonization can be preferred to binary skeletonization [23, 1, 9, 13]: fingerprint analysis, medical image processing and optical character recognition.

Many fingerprint analysis systems use skeletonization as an essential step. Usually, the fingerprint image is binarized before skeletonization. Here, we present a way to obtain a (binary) skeleton without a prior binarization of the image (see figure 4). After a grayscale thinning (b), we use the remaining gray scale information to select robust crest points (c) having high contrast with their background (d).



The crest points are formally defined as follows. Let  $\alpha$  be an integer, we say that a point  $x$  is a *crest point with contrast  $\alpha$*  for an image  $F$  if there exists a level  $k$  such that  $\overline{T}(x, F_k) \geq 2$ , and such that  $k - \max\{F(y), y \in \overline{F_k} \cap \mathcal{N}_0^*(x)\} \geq \alpha$ . For example in figure 3(d), the points at levels 8 and 9 are not crest points with contrast  $\alpha = 10$  for example, but they are crest points with contrast  $\alpha = 2$ . In figure 5(d) we show the crest points with contrast  $\alpha = 50$ . As we can see the resulting skeleton is free of spurious branches and is well centered.

We illustrate two other applications in figure 5. The first is the thinning of a vascular network in an image of a human retina. The vessels correspond to the lighter pixels in figure 5(a). After the gray scale thinning, we obtain the image in (b). A second application is the thinning of scanned characters shown in figures 5(c) and (d).

## 5 Conclusion

In this paper, we introduced a parallel thinning algorithm and proved its topological soundness, using some properties issued from the framework of critical kernels. We also sketched some possible applications, in areas where the benefits of avoiding segmentation prior to skeletonization have been pointed out by several authors. The perspectives of this work include: the analysis of the computational cost of our algorithm, both in theory and in practice; the introduction and study of an asymmetric parallel thinning algorithm, evoked in the previous section; the introduction and study of a faster algorithm dedicated to the case of ultimate thinning; the validation of this approach by its evaluation in the context of a real-world application. These items will be developed in a forthcoming paper.

## References

1. Abeysinghe, S.S., Baker, M., Chiu, W., Ju, T.: Segmentation-free skeletonization of grayscale volumes for shape understanding. In: IEEE International Conference on Shape Modeling and Applications, SMI 2008, pp. 63–71 (2008)
2. Arcelli, C., Ramella, G.: Finding grey-skeletons by iterated pixel removal. *Image and Vision Computing* 13(3), 159–167 (1995)
3. Bertrand, G.: On critical kernels. *Comptes Rendus de l'Académie des Sciences, Série Math.* I(345), 363–367 (2007)
4. Bertrand, G., Couprie, M.: Two-dimensional thinning algorithms based on critical kernels. *J. of Mathematical Imaging and Vision* 31(1), 35–56 (2008)
5. Bertrand, G., Couprie, M.: Powerful parallel and symmetric 3d thinning schemes based on critical kernels. *J. of Mathematical Imaging and Vision* (2012) (to appear), doi:10.1007/s10851-012-0402-7
6. Bertrand, G., Everat, J.C., Couprie, M.: Topological approach to image segmentation. In: *SPIE Vision Geometry V*, vol. 2826, pp. 65–76 (1996)
7. Bertrand, G., Everat, J.C., Couprie, M.: Image segmentation through operators based upon topology. *J. of Electronic Imaging* 6(4), 395–405 (1997)
8. Beucher, S.: Segmentation d'images et morphologie mathématique. PhD thesis, École des Mines de Paris, France (1990)

9. Costes, C., Garello, R., Mercier, G., Artis, J.-P., Bon, N.: Convective clouds modelling and tracking by an airborne radar. In: OCEANS 2008, pp. 1–5 (September 2008)
10. Couprie, M., Bezerra, F.N., Bertrand, G.: Topological operators for grayscale image processing. *J. of Electronic Imaging* 10(4), 1003–1015 (2001)
11. Dyer, C., Rosenfeld, A.: Thinning algorithms for gray-scale pictures. *IEEE Trans. on Pattern An. and Machine Int.* 1(1), 88–89 (1979)
12. Jang, J.-H., Hong, K.-S.: A pseudo-distance map for the segmentation-free skeletonization of gray-scale images. In: Proceedings of the Eighth IEEE International Conference on Computer Vision, ICCV 2001, vol. 2, pp. 18–23 (2001)
13. Kang, K.W., Suh, J.W., Kim, J.H.: Skeletonization of grayscale character images using pixel superiority index. In: IAPR Workshop on Document Analysis Systems (1998)
14. Kong, T.Y.: On topology preservation in 2D and 3D thinning. *Int. J. on Pattern Recognition and Artificial Intelligence* 9, 813–844 (1995)
15. Kong, T.Y., Rosenfeld, A.: Digital topology: introduction and survey. *Computer Vision, Graphics and Image Processing* 48, 357–393 (1989)
16. Le Bourgeois, F., Emptoz, H.: Skeletonization by gradient diffusion and regularization. In: IEEE International Conference on Image Processing, ICIP 2007, vol. 3, pp. 33–36 (2007)
17. Lohou, C., Bertrand, G.: New parallel thinning algorithms for 2d grayscale images. In: Latecki, L.J., Mount, D.M., Wu, A.Y. (eds.) Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, vol. 4117, pp. 58–69 (2000)
18. Mersa, S.S., Darwish, A.M.: A new parallel thinning algorithm for gray scale images. In: IEEE Nonlinear Signal and Image Proc. Conf., pp. 409–413 (1999)
19. Nedzved, A., Uchida, S., Ablameyko, S.: Gray-scale thinning by using a pseudo-distance map. In: 18th International Conference on Pattern Recognition, ICPR 2006, vol. 2, pp. 239–242 (2006)
20. Pal, S.K.: Fuzzy skeletonization of an image. *Pattern Recognition Letters* 10, 17–23 (1989)
21. Ranwez, V., Soille, P.: Order independent homotopic thinning for binary and grey tone anchored skeletons. *Pattern Recognition Letters* 23(6), 687–702 (2002); *Discrete Geometry for Computer Imagery*
22. Rosenfeld, A.: The fuzzy geometry of image subsets. *Pattern Recognition Letters* 2, 311–317 (1984)
23. Saleh, A.M., Bahaa Eldin, A.M., Wahdan, A.-M.A.: A modified thinning algorithm for fingerprint identification systems. In: International Conference on Computer Engineering Systems, ICCES 2009, pp. 371–376 (2009)
24. Whitehead, J.H.C.: Simplicial spaces, nuclei and  $m$ -groups. *Proceedings of the London Mathematical Society* 45(2), 243–327 (1939)
25. Yu, S.-S., Tsai, W.-H.: A new thinning algorithm for gray-scale images by the relaxation technique. *Pattern Recognition* 23(10), 1067–1076 (1990)

# New Structures Based on Completions\*

Gilles Bertrand

Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge  
Equipe A3SI, ESIEE Paris

**Abstract.** We propose new axioms relative to combinatorial topology. These axioms are settled in the framework of completions which are inductive properties expressed in a declarative way, and that may be combined.

We introduce several completions for describing *dyads*. A dyad is a pair of complexes which are, in a certain sense, linked by a “relative topology”.

We first give some basic properties of dyads, then we introduce a second set of axioms for *relative dendrites*. This allows us to establish a theorem which provides a link between dyads and dendrites, a dendrite is an acyclic complex which may be also described by completions. Thanks to a previous result, this result makes clear the relation between dyads, relative dendrites, and complexes which are acyclic in the sense of homology.

**Keywords:** Acyclic complexes, Combinatorial topology, Simplicial Complexes, Collapse, Completions.

## 1 Introduction

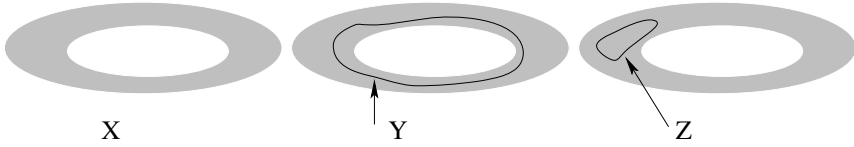
Simple homotopy plays a fundamental role in combinatorial topology [1–7]. It has also been shown that the collapse operation is fundamental to interpret some notions relative to homotopy in the context of computer imagery [8–10], see also [11–13].

In this paper, we further investigate an axiomatic approach related to simple homotopy. This approach has been introduced in [14] where the notion of a *dendrite* was presented through two simple axioms. A dendrite is an acyclic object. A theorem asserts that an object is a dendrite if and only if it is acyclic in the sense of homology.

Here, we present new axioms for describing *dyads*. Intuitively, a dyad is a couple of objects  $(X, Y)$ , with  $X \subseteq Y$ , such that the cycles of  $X$  are “at the right place with respect to the ones of  $Y$ ”. Let us consider Fig. 1, where an object  $X$ , and two objects  $Y \subseteq X$ ,  $Z \subseteq X$  are depicted. We see that it is possible to continuously deform  $Y$  onto  $X$ , this deformation keeping  $Y$  inside  $X$ . Thus, the pair  $(Y, X)$  is a dyad. On the other hand,  $Z$  is homotopic to  $X$ , but  $Z$  is not “at the right place”, therefore  $(Z, X)$  is not a dyad.

---

\* This work has been partially supported by the “ANR-2010-BLAN-0205 KIDICO” project.



**Fig. 1.** An object  $X$  (an annulus), and two objects  $Y \subseteq X$ ,  $Z \subseteq X$  (two simple closed curves). The pair  $(Y, X)$  is a dyad, while  $(Z, X)$  is not.

The paper is organized as follows. First, we give some basic definitions for simplicial complexes (Sec. 2). Then, we recall some basic facts relative to the notion of a completion (Sec. 3), completions will be used as a language for describing our axioms. We also recall the definition of a dendrite (Sec. 4). In the two following sections we introduce new axioms for presenting the notion of a dyad (Sec. 5), and the notion of a relative dendrite (Sec. 6). In Sec. 7, we give a theorem (Th. 4) which makes clear the link between dyads and dendrites. Thanks to a previous result, this result makes clear the relation between dyads, relative dendrites, and complexes which are acyclic in the sense of homology.

The paper is self contained. In particular, almost all proofs are included.

## 2 Basic Definitions for Simplicial Complexes

Let  $X$  be a finite family composed of finite sets. The *simplicial closure* of  $X$  is the complex  $X^- = \{y \subseteq x \mid x \in X\}$ . The family  $X$  is a (*finite simplicial*) *complex* if  $X = X^-$ . We write  $\mathbb{S}$  for the collection of all finite simplicial complexes. Note that  $\emptyset \in \mathbb{S}$  and  $\{\emptyset\} \in \mathbb{S}$ ,  $\emptyset$  is the *void complex*, and  $\{\emptyset\}$  is the *empty complex*.

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  $Y \preceq X$ .

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  $-1$ .

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. A cell  $\alpha \in \mathbb{C}$  is a *vertex* if  $\dim(\alpha) = 0$ .

The *ground set* of  $X \in \mathbb{S}$  is the set  $\underline{X} = \cup\{x \in X \mid \dim(x) = 0\}$ . We say that  $X \in \mathbb{S}$  and  $Y \in \mathbb{S}$  are *disjoint*, or that  $X$  is *disjoint from*  $Y$ , if  $\underline{X} \cap \underline{Y} = \emptyset$ . Thus,  $X$  and  $Y$  are disjoint if and only if  $X \cap Y = \emptyset$  or  $X \cap Y = \{\emptyset\}$ .

If  $X \in \mathbb{S}$  and  $Y \in \mathbb{S}$  are disjoint, the *join* of  $X$  and  $Y$  is the simplicial 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  $Y = \{\emptyset\}$ . The join  $\alpha X$  of a vertex  $\alpha$  and a complex  $X \in \mathbb{S}$  is a *cone*.

**Important Convention.** In this paper, if  $X, Y \in \mathbb{S}$ , we implicitly assume that  $X$  and  $Y$  have disjoint ground sets whenever we write  $XY$ .

Let  $A \in \mathbb{C}$  and  $X \preceq A$ . The dual of  $X$  for  $A$  is the simplicial complex, written  $X_A^*$ , such that  $X_A^* = \{x \in A \mid (\underline{A} \setminus x) \notin X\}$ .

We have  $\emptyset_A^* = A$  and  $\{\emptyset\}_A^* = A^\circ$ , and, for any  $A \in \mathbb{C}$ , we have the following:

- If  $X \preceq A$ , then  $(X_A^*)^*_A = X$ .
- If  $X \preceq A, Y \preceq A$ , then  $(X \cup Y)_A^* = X_A^* \cap Y_A^*$  and  $(X \cap Y)_A^* = X_A^* \cup Y_A^*$ .

### 3 Completions

We give some basic definitions for completions, they will allow us to formulate our axioms as well as to combine them. A completion may be seen as a rewriting rule which permits to derive collections of sets. See [14] for more details.

Let  $\mathbf{S}$  be a given collection and let  $\mathcal{K}$  be an arbitrary subcollection of  $\mathbf{S}$ . Thus, we have  $\mathcal{K} \subseteq \mathbf{S}$ . In the sequel of the paper, the symbol  $\mathcal{K}$ , with possible superscripts, will be a dedicated symbol (a kind of variable).

Let  $\kappa$  be a binary relation on  $2^{\mathbf{S}}$ , thus  $\kappa \subseteq 2^{\mathbf{S}} \times 2^{\mathbf{S}}$ . We say that  $\kappa$  is *finitary*, if  $\mathbf{F}$  is finite whenever  $(\mathbf{F}, \mathbf{G}) \in \kappa$ .

Let  $\langle \mathbf{K} \rangle$  be a property which depends on  $\mathcal{K}$ . We say that  $\langle \mathbf{K} \rangle$  is a *completion (on  $\mathbf{S}$ )* if  $\langle \mathbf{K} \rangle$  may be expressed as the following property:

$\rightarrow$  If  $\mathbf{F} \subseteq \mathcal{K}$ , then  $\mathbf{G} \subseteq \mathcal{K}$  whenever  $(\mathbf{F}, \mathbf{G}) \in \kappa$ .  $\langle \kappa \rangle$

where  $\kappa$  is an arbitrary finitary binary relation on  $2^{\mathbf{S}}$ .

If  $\langle \mathbf{K} \rangle$  is 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  $\mathcal{K} = \mathbf{X}$ .

**Theorem 1.** [14] *Let  $\langle \mathbf{K} \rangle$  be a completion 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  $\langle \mathbf{K} \rangle$ .*

If  $\langle \mathbf{K} \rangle$  is a completion on  $\mathbf{S}$  and if  $\mathbf{X} \subseteq \mathbf{S}$ , we write  $\langle \mathbf{X}; \mathbf{K} \rangle$  for the unique minimal collection which contains  $\mathbf{X}$  and which satisfies  $\langle \mathbf{K} \rangle$ .

Let  $\langle \mathbf{K} \rangle$  be a completion which is expressed as the above property  $\langle \kappa \rangle$ . By a fixed point property, the collection  $\langle \mathbf{X}; \mathbf{K} \rangle$  may be obtained by starting from  $\mathcal{K} = \mathbf{X}$ , and by iteratively adding to  $\mathcal{K}$ , until idempotence, all the sets  $\mathbf{G}$  such that  $(\mathbf{F}, \mathbf{G}) \in \kappa$  and  $\mathbf{F} \subseteq \mathcal{K}$  (see [14]).

Let  $\langle \mathbf{K} \rangle$  and  $\langle \mathbf{Q} \rangle$  be two completions on  $\mathbf{S}$ . It may be seen that  $\langle \mathbf{K} \rangle \wedge \langle \mathbf{Q} \rangle$  is a completion, the symbol  $\wedge$  standing for the logical “and”. In the sequel of the paper, we write  $\langle \mathbf{K}, \mathbf{Q} \rangle$  for  $\langle \mathbf{K} \rangle \wedge \langle \mathbf{Q} \rangle$ . Also, if  $\mathbf{X} \subseteq \mathbf{S}$ , the notation  $\langle \mathbf{X}; \mathbf{K}, \mathbf{Q} \rangle$  stands for the smallest collection which contains  $\mathbf{X}$  and which satisfies  $\langle \mathbf{K} \rangle \wedge \langle \mathbf{Q} \rangle$ .

**Example.** Let us consider the collection  $\mathbf{S} = \mathbb{S}$ . Thus,  $\mathcal{K}$  denotes an arbitrary collection of simplicial complexes. We define the property  $\langle \mathcal{T} \rangle$  as follows:

$\rightarrow$  If  $S, T \in \mathcal{K}$ , then  $S \cup T \in \mathcal{K}$  whenever  $S \cap T \neq \{\emptyset\}$ .  $\langle \mathcal{T} \rangle$

Let  $\kappa$  be the binary relation on  $2^{\mathbf{S}}$  such that  $(\mathbf{F}, \mathbf{G}) \in \kappa$  iff there exist  $S, T \in \mathbb{S}$ , with  $\mathbf{F} = \{S, T\}$ ,  $\mathbf{G} = \{S \cup T\}$ , and  $S \cap T \neq \{\emptyset\}$ . We see that  $\kappa$  is finitary and that  $\langle \mathcal{T} \rangle$  may be expressed as the property  $\langle \kappa \rangle$ . Thus  $\langle \mathcal{T} \rangle$  is a completion. Now, let us consider the collection  $\Pi = \langle \mathbb{C}; \mathcal{T} \rangle$ . It may be checked that  $\Pi$  is precisely the collection of all simplicial complexes which are (path) connected

(see also [17] where the property  $\langle \mathcal{T} \rangle$  is used in a different context). Having in mind the above iterative procedure,  $\langle \mathbb{C}, \mathcal{T} \rangle$  may be seen as a dynamic structure where the completion  $\langle \mathcal{T} \rangle$  acts as a generator, which, from  $\mathbb{C}$ , makes it possible to enumerate all finite connected simplicial complexes.

### 4 Dendrites

The notion of a dendrite was introduced in [14] as a way for defining a remarkable collection made of acyclic complexes.

In the rest of the paper,  $\mathcal{K}$  will denote an arbitrary subcollection of  $\mathbb{S}$ .

**Definition 1.** We define the two completions on  $\mathbb{S}$ : For any  $S, T \in \mathbb{S}$ ,

$$\rightarrow \text{If } S, T \in \mathcal{K}, \text{ then } S \cup T \in \mathcal{K} \text{ whenever } S \cap T \in \mathcal{K}. \tag{D1}$$

$$\rightarrow \text{If } S, T \in \mathcal{K}, \text{ then } S \cap T \in \mathcal{K} \text{ whenever } S \cup T \in \mathcal{K}. \tag{D2}$$

We set  $\mathbb{R} = \langle \mathbb{C}; \text{D1} \rangle$  and  $\mathbb{D} = \langle \mathbb{C}; \text{D1}, \text{D2} \rangle$ , thus we have  $\mathbb{R} \subseteq \mathbb{D}$ .

Each element of  $\mathbb{R}$  is a ramification and each element of  $\mathbb{D}$  is a dendrite.

Let us recall some basic definitions relative to simple homotopy [1], note that these notions may also be introduced by the means of completions [14].

Let  $X, Y \in \mathbb{S}$  and  $x, y$  be two distinct faces of  $X$ . If  $y$  is the only face of  $X$  which contains  $x$ , then  $Y = X \setminus \{x, y\}$  is an elementary collapse of  $X$ . We say that  $X$  collapses onto  $Y$ , if there exists a sequence  $\langle X_0, \dots, X_k \rangle$  such that  $X_0 = X$ ,  $X_k = Y$ , and  $X_i$  is an elementary collapse of  $X_{i-1}$ ,  $i \in [1, k]$ . The complex  $X$  is collapsible if  $X$  collapses onto  $\emptyset$ . We say that  $X$  is (simple) homotopic to  $Y$  if there exists a sequence  $\langle X_0, \dots, X_k \rangle$  such that  $X_0 = X$ ,  $X_k = Y$ , and either  $X_i$  is an elementary collapse of  $X_{i-1}$ , or  $X_{i-1}$  is an elementary collapse of  $X_i$ ,  $i \in [1, k]$ . The complex  $X$  is (simply) contractible if  $X$  is simple homotopic to  $\emptyset$ .

For example, if  $X$  is a tree, then  $X$  is collapsible,  $X$  is a dendrite, and also a ramification. In fact, any collapsible complex is a ramification [6]. The Bing's house with two rooms [15] and the dunce hat [16] are classical examples of complexes which are contractible but not collapsible. Both of them are dendrites.

In fact, it was shown [14] that any simply contractible complex is a dendrite. Furthermore it was shown that:

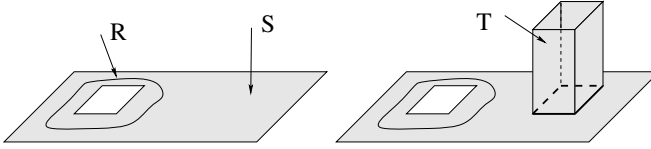
- a complex is a dendrite if and only if it is acyclic in the sense of homology; and
- a complex is a dendrite if and only if its suspension is simply contractible.

### 5 Dyads

In this section, we introduce the notion of a dyad and give some propositions which are necessary to establish one of the main result of the paper (Th. 4). See the introduction and Fig. 1 for an intuitive presentation of a dyad. See also Fig. 2 for an illustration of the axiom  $\langle \check{X}1 \rangle$ .

$$\text{We set } \check{\mathbb{S}} = \{(X, Y) \mid X, Y \in \mathbb{S}, X \preceq Y\} \text{ and } \check{\mathbb{C}} = \{(A, B) \in \check{\mathbb{S}} \mid A, B \in \mathbb{C}\}.$$

In the sequel of the paper,  $\check{\mathcal{K}}$  will denote an arbitrary subcollection of  $\check{\mathbb{S}}$ . Furthermore,  $\alpha$  and  $\beta$  will always denote vertices.



**Fig. 2.** Two objects  $R, S$  which constitute a dyad  $(R, S)$ . An object  $T$  which is glued to  $S$ . The couple  $(S \cap T, T)$  is a dyad, thus, by axiom  $\langle \check{X}1 \rangle$ ,  $(R, S \cup T)$  is also a dyad.

**Definition 2.** We define three completions on  $\check{\mathbb{S}}$ : For any  $(R, S) \in \check{\mathbb{S}}, T \in \mathbb{S}$ ,

- > If  $(R, S) \in \check{\mathcal{K}}$  and  $(S \cap T, T) \in \check{\mathcal{K}}$ , then  $(R, S \cup T) \in \check{\mathcal{K}}$ .  $\langle \check{X}1 \rangle$
- > If  $(R, S) \in \check{\mathcal{K}}$  and  $(R, S \cup T) \in \check{\mathcal{K}}$ , then  $(S \cap T, T) \in \check{\mathcal{K}}$ .  $\langle \check{X}2 \rangle$
- > If  $(R, S \cup T) \in \check{\mathcal{K}}$  and  $(S \cap T, T) \in \check{\mathcal{K}}$ , then  $(R, S) \in \check{\mathcal{K}}$ .  $\langle \check{X}3 \rangle$

We set  $\check{\mathbb{X}} = \langle \check{\mathbb{C}}; \check{X}1, \check{X}2, \check{X}3 \rangle$ . Each element of  $\check{\mathbb{X}}$  is a dyad.

We introduce the following completions on  $\check{\mathbb{S}}$  (the symbols  $\check{\mathbb{T}}, \check{\mathbb{U}}, \check{\mathbb{L}}$  stand respectively for “transitivity”, “upper confluence”, and “lower confluence”):  
 For any  $(R, S), (S, T), (R, T) \in \check{\mathbb{S}}$ ,

- > If  $(R, S) \in \check{\mathcal{K}}$  and  $(S, T) \in \check{\mathcal{K}}$ , then  $(R, T) \in \check{\mathcal{K}}$ .  $\langle \check{\mathbb{T}} \rangle$
- > If  $(R, S) \in \check{\mathcal{K}}$  and  $(R, T) \in \check{\mathcal{K}}$ , then  $(S, T) \in \check{\mathcal{K}}$ .  $\langle \check{\mathbb{U}} \rangle$
- > If  $(R, T) \in \check{\mathcal{K}}$  and  $(S, T) \in \check{\mathcal{K}}$ , then  $(R, S) \in \check{\mathcal{K}}$ .  $\langle \check{\mathbb{L}} \rangle$

Considering complexes  $R, S, T$  such that  $R \preceq S \preceq T$ , we see that we obtain directly  $\langle \check{\mathbb{T}} \rangle, \langle \check{\mathbb{U}} \rangle, \langle \check{\mathbb{L}} \rangle$  from  $\langle \check{X}1 \rangle, \langle \check{X}2 \rangle, \langle \check{X}3 \rangle$ , respectively. Thus, we have:

**Proposition 1.** The collection  $\check{\mathbb{X}}$  satisfies the properties  $\langle \check{\mathbb{T}} \rangle, \langle \check{\mathbb{U}} \rangle$ , and  $\langle \check{\mathbb{L}} \rangle$ .

**Proposition 2.** For any  $X \in \mathbb{S}$ , we have  $(\emptyset, \alpha X) \in \check{\mathbb{X}}$ .

**Proof.** If  $X = \emptyset$ , then  $(\emptyset, \alpha X) \in \check{\mathbb{X}}$  (since  $(\emptyset, \emptyset) \in \check{\mathbb{C}}$ ). If  $X = \{\emptyset\}$ , then  $(\emptyset, \alpha X) \in \check{\mathbb{X}}$  (since  $\alpha X = \alpha$  and  $(\emptyset, \alpha) \in \check{\mathbb{C}}$ ). Suppose  $X \neq \emptyset$  and  $X \neq \{\emptyset\}$ .

i) If  $X$  has a single facet, then  $X \in \mathbb{C}$ . Thus  $(\emptyset, \alpha X) \in \check{\mathbb{X}}$  (since  $(\emptyset, \alpha X) \in \check{\mathbb{C}}$ );

ii) If  $X$  has more than one facet, then there exists  $X', X'' \in \mathbb{S}$  such that  $X = X' \cup X''$ , and  $Card(X') < Card(X), Card(X'') < Card(X)$ . Suppose that  $(\emptyset, \alpha X') \in \check{\mathbb{X}}, (\emptyset, \alpha X'') \in \check{\mathbb{X}}$ , and  $(\emptyset, \alpha(X' \cap X'')) \in \check{\mathbb{X}}$ . Then, by  $\langle \check{\mathbb{U}} \rangle$ , we have  $(\alpha(X' \cap X''), \alpha X'') \in \check{\mathbb{X}}$ . Therefore, by  $\langle \check{X}1 \rangle$  (setting  $R = \emptyset, S = \alpha X', T = \alpha X''$ ), we have  $(\emptyset, \alpha X) \in \check{\mathbb{X}}$ . The result follows by induction on  $Card(X)$ .  $\square$

**Proposition 3.** For any  $X \in \mathbb{S}$ , we have  $(X, X) \in \check{\mathbb{X}}$ .

**Proof.** By Prop. 2, we have  $(\emptyset, \alpha X) \in \check{\mathbb{X}}$ . Since  $\check{\mathbb{X}}$  satisfies  $\langle \check{\mathbb{U}} \rangle$ , it implies that  $(\alpha X, \alpha X) \in \check{\mathbb{X}}$  (setting  $R = \emptyset$ , and  $S = T = \alpha X$ ). By  $\langle \check{X}2 \rangle$  (setting  $R = S = \alpha X$ , and  $T = X$ ), this gives  $(\alpha X \cap X, X) = (X, X) \in \check{\mathbb{X}}$ .  $\square$

We define two completions on  $\check{\mathbb{S}}$ : For any  $S, T \in \mathbb{S}$ ,

- > If  $(S \cap T, T) \in \check{\mathcal{K}}$ , then  $(S, S \cup T) \in \check{\mathcal{K}}$ .  $\langle \check{Y}1 \rangle$
- > If  $(S, S \cup T) \in \check{\mathcal{K}}$ , then  $(S \cap T, T) \in \check{\mathcal{K}}$ .  $\langle \check{Y}2 \rangle$

We give, hereafter, a theorem (Th. 2) which provides another way to generate the collection  $\check{\mathbb{X}}$ . This theorem will be used in section 7 to establish a link between dendrites and dyads. Before, we make a remark on a basic property of completions which allows one to establish the equivalence between two completions structures. This property is necessary for the proof of Th. 2.

*Remark 1.* Let  $\langle K \rangle$  be a completion on  $\mathbf{S}$  and let  $\mathbf{X} \subseteq \mathbf{S}$ . It may be shown [14] that we have  $\langle \mathbf{X}; K \rangle = \cap \{ \mathbf{Y} \subseteq \mathbf{S} \mid \mathbf{X} \subseteq \mathbf{Y} \text{ and } \mathbf{Y} \text{ satisfies } \langle K \rangle \}$ . Thus, if a given collection  $\mathbf{Y} \subseteq \mathbf{S}$  is such that  $\mathbf{X} \subseteq \mathbf{Y}$  and  $\mathbf{Y}$  satisfies  $\langle K \rangle$ , then we have necessarily  $\langle \mathbf{X}; K \rangle \subseteq \mathbf{Y}$ .

**Theorem 2.** *We have  $\check{\mathbb{X}} = \langle \check{\mathbb{C}}; \check{Y}1, \check{Y}2, \check{T}, \check{U}, \check{L} \rangle$ .*

**Proof.** We set  $\check{\mathbb{X}}' = \langle \check{\mathbb{C}}; \check{Y}1, \check{Y}2, \check{T}, \check{U}, \check{L} \rangle$ . As a consequence of Prop. 3, we can obtain  $\langle \check{Y}1 \rangle$  and  $\langle \check{Y}2 \rangle$  from  $\langle \check{X}1 \rangle$  and  $\langle \check{X}2 \rangle$ , respectively (setting  $R = S$ ). The collection  $\check{\mathbb{X}}$  also satisfies the properties  $\langle \check{T} \rangle$ ,  $\langle \check{U} \rangle$ ,  $\langle \check{L} \rangle$  (Prop. 1). Thus, since  $\check{\mathbb{C}} \subseteq \check{\mathbb{X}}$ , we have  $\check{\mathbb{X}}' \subseteq \check{\mathbb{X}}$  (see remark 1). Now, let  $(R, S) \in \check{\mathbb{S}}$  and  $T \in \mathbb{S}$ :

- Suppose  $(R, S) \in \check{\mathbb{X}}'$  and  $(S \cap T, T) \in \check{\mathbb{X}}'$ . Then, by  $\langle \check{Y}1 \rangle$ , we have  $(S, S \cup T) \in \check{\mathbb{X}}'$ . Therefore, by  $\langle \check{T} \rangle$ , we have  $(R, S \cup T) \in \check{\mathbb{X}}'$ ,
- Suppose  $(R, S) \in \check{\mathbb{X}}'$  and  $(R, S \cup T) \in \check{\mathbb{X}}'$ . Then, by  $\langle \check{U} \rangle$ , we have  $(S, S \cup T) \in \check{\mathbb{X}}'$ . Therefore, by  $\langle \check{Y}2 \rangle$ , we have  $(S \cap T, T) \in \check{\mathbb{X}}'$ ,
- Suppose  $(R, S \cup T) \in \check{\mathbb{X}}'$  and  $(S \cap T, T) \in \check{\mathbb{X}}'$ . Then, by  $\langle \check{Y}1 \rangle$ ,  $(S, S \cup T) \in \check{\mathbb{X}}'$ . Therefore, by  $\langle \check{L} \rangle$ , we have  $(R, S) \in \check{\mathbb{X}}'$ .

It follows that  $\check{\mathbb{X}}'$  satisfies the three properties  $\langle \check{X}1 \rangle$ ,  $\langle \check{X}2 \rangle$ ,  $\langle \check{X}3 \rangle$ . Thus, since  $\check{\mathbb{C}} \subseteq \check{\mathbb{X}}'$ , we have  $\check{\mathbb{X}} \subseteq \check{\mathbb{X}}'$  (see remark 1). □

## 6 Relative Dendrites

In this section, we introduce new axioms for defining the notion of a relative dendrite. We will see in the sequel (next section) that these axioms provide another way to describe dyads. We set  $\check{\mathbb{C}}^+ = \check{\mathbb{C}} \cup \{(\{\emptyset\}, \{\emptyset\})\}$ .

**Definition 3.** *We define two completions on  $\check{\mathbb{S}}$ : For any  $(S, T), (S', T') \in \check{\mathbb{S}}$ ,*

- > *If  $(S, T), (S', T'), (S \cap S', T \cap T') \in \check{\mathcal{K}}$ , then  $(S \cup S', T \cup T') \in \check{\mathcal{K}}$ .*  $\langle \check{Z}1 \rangle$

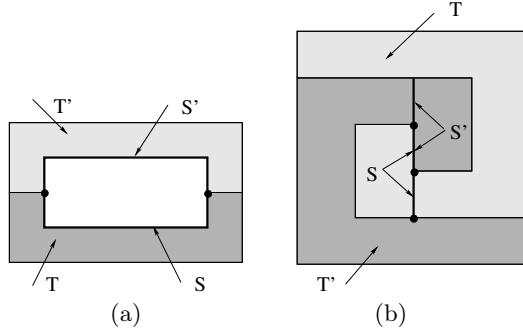
- > *If  $(S, T), (S', T'), (S \cup S', T \cup T') \in \check{\mathcal{K}}$ , then  $(S \cap S', T \cap T') \in \check{\mathcal{K}}$ .*  $\langle \check{Z}2 \rangle$

*Each element of  $\langle \check{\mathbb{C}}^+; \check{Z}1, \check{Z}2 \rangle$  is called a relative dendrite.*

In Fig. 3, two examples of two couples  $(S, T), (S', T') \in \check{\mathbb{S}}$  which satisfy the conditions of  $\langle \check{Z}1 \rangle$  are given. Thus, in these two examples,  $(S \cup S', T \cup T')$  is a relative dendrite.

In Fig. 3 (a),  $(S \cup S', T \cup S')$  and  $(T \cup S', T \cup T')$  are dyads (this fact may be seen using the forthcoming Prop. 9). Then, using  $\langle \check{T} \rangle$ , it is possible to generate  $(S \cup S', T \cup T')$  with the axioms of a dyad.





**Fig. 3.** (a) and (b): Two examples of two couples  $(S, T), (S', T') \in \check{\mathbb{S}}$  which satisfy the conditions of  $\langle \check{Z}1 \rangle$  (we consider triangulations of these objects). In (a),  $S$  and  $S'$  are two simple open curves,  $S \cap S'$  is a complex made of two vertices. In (b),  $S$  and  $S'$  are also two simple open curves, but  $S \cap S'$  is a complex made of a segment.

Now, we observe that, in Fig. 3 (b),  $(S \cup S', T \cup T')$  is not a dyad (it can be checked that  $X$  and  $Y$  must have the same Euler characteristic whenever  $(X, Y)$  is a dyad). Thus, it is not possible to generate, in a straightforward manner, the relative dendrite  $(S \cup S', T \cup T')$  with the axioms of a dyad.

*Remark 2.* As a direct consequence of the definitions of  $\langle \check{Z}1 \rangle, \langle \check{Z}2 \rangle$ , and the one of a dendrite, we have  $\langle \check{C}; \check{Z}1, \check{Z}2 \rangle \subseteq \{(X, Y) \in \check{\mathbb{S}} \mid X \in \mathbb{D}, Y \in \mathbb{D}\}$ . This fact emphasizes the role of  $(\{\emptyset\}, \{\emptyset\})$  in  $\langle \check{C}^+; \check{Z}1, \check{Z}2 \rangle$ .

Let  $(X, Y) \in \check{\mathbb{S}}$ . If  $\alpha$  is a vertex such that  $\alpha X \cap Y = X$ , we say that  $\alpha X \cup Y$  is a *cone on*  $(X, Y)$ , and we write  $\alpha X \ddot{\cup} Y$  for  $\alpha X \cup Y$ .

**Proposition 4.** *Let  $Z \in \mathbb{S}$  and let  $\alpha$  be an arbitrary vertex. There exists a unique couple  $(X, Y) \in \check{\mathbb{S}}$  such that  $Z = \alpha X \ddot{\cup} Y$ .*

Thus, by Prop. 4, if  $Z \in \mathbb{S}$  and if  $\alpha$  is an arbitrary vertex, the complexes  $X$  and  $Y$  are specified whenever we write  $Z = \alpha X \ddot{\cup} Y$ . Note that we may have  $\alpha \not\leq Z$ , in this case  $X = \emptyset$  and  $Z = Y$ .

**Proposition 5.** *Let  $Z, Z', Z'' \in \mathbb{S}$ .*

*We set  $Z = \alpha X \ddot{\cup} Y, Z' = \alpha X' \ddot{\cup} Y',$  and  $Z'' = \alpha X'' \ddot{\cup} Y''$ .*

- 1) *If  $Z = Z' \cup Z''$ , then  $X = X' \cup X''$  and  $Y = Y' \cup Y''$ ;*
- 2) *If  $Z = Z' \cap Z''$ , then  $X = X' \cap X''$  and  $Y = Y' \cap Y''$ .*

**Proof.** The result follows from 1), 2), and Prop. 4.

1) If  $Z = Z' \cup Z''$ , then  $Z = \alpha(X' \cup X'') \cup (Y' \cup Y'')$ . Furthermore, since  $(X' \cup X'') \subseteq (Y' \cup Y'')$  and since  $\alpha$  is disjoint from  $Y' \cup Y''$ , we have  $\alpha(X' \cup X'') \cap (Y' \cup Y'') = X' \cup X''$ .

2) Suppose  $Z = Z' \cap Z''$ . Then  $Z = (\alpha X' \cup Y') \cap (\alpha X'' \cup Y'') = \alpha(X' \cap X'') \cup (Y' \cap Y'') \cup (\alpha X' \cap Y'') \cup (\alpha X'' \cap Y')$ . Since  $(\alpha X' \cap Y'') \cup (\alpha X'' \cap Y') \subseteq Y' \cap Y''$ , we have  $Z = \alpha(X' \cap X'') \cup (Y' \cap Y'')$ . Furthermore, since  $(X' \cap X'') \subseteq (Y' \cap Y'')$  and since  $\alpha$  is disjoint from  $Y' \cap Y''$ , we have  $\alpha(X' \cap X'') \cap (Y' \cap Y'') = X' \cap X''$ .  $\square$

**Theorem 3.** *Let  $(X, Y) \in \ddot{\mathbb{S}}$ . The couple  $(X, Y)$  is a relative dendrite if and only if  $\alpha X \ddot{\cup} Y$  is a dendrite.*

**Proof**

1) If  $(X, Y) \in \ddot{\mathbb{C}}^+$ , we see that  $\alpha X \ddot{\cup} Y$  is a ramification. Thus,  $\alpha X \ddot{\cup} Y$  is a dendrite. Suppose  $R = \alpha S \ddot{\cup} T$  and  $R' = \alpha S' \ddot{\cup} T'$  are dendrites. Then, by the very definition of a dendrite,  $R \cap R'$  is a dendrite if and only if  $R \cup R'$  is a dendrite. Consequently, by Prop. 5,  $\alpha(S \cap S') \ddot{\cup} (T \cap T')$  is a dendrite if and only if  $\alpha(S \cup S') \ddot{\cup} (T \cup T')$  is a dendrite. By the preceding remarks, we may affirm, by induction on  $\langle \ddot{\mathbb{C}}^+; \check{Z}1, \check{Z}2 \rangle$ , that  $\alpha X \ddot{\cup} Y$  is a dendrite whenever  $(X, Y)$  is a relative dendrite.

2) Suppose  $Z = \alpha X \ddot{\cup} Y$  is a dendrite.

i) Suppose  $Z \in \mathbb{C}$ . If  $X = \emptyset$ , we have  $(X, Y) \in \ddot{\mathbb{C}}$ . If  $X = \{\emptyset\}$ , we must have  $Y = \{\emptyset\}$ , otherwise  $Z$  would not be connected, thus  $(X, Y) \in \ddot{\mathbb{C}}^+$ . If  $X \neq \emptyset$  and  $X \neq \{\emptyset\}$ , it may be seen that we must have  $X \in \mathbb{C}$  and  $Y = X$ , thus  $(X, Y) \in \ddot{\mathbb{C}}$ .

ii) Suppose we have  $Z = Z' \cup Z''$ , with  $Z', Z'', Z' \cap Z'' \in \mathbb{D}$ . We set  $Z' = \alpha X' \ddot{\cup} Y'$  and  $Z'' = \alpha X'' \ddot{\cup} Y''$ . If  $(X', Y'), (X'', Y''), (X' \cap X'', Y' \cap Y'')$  are relative dendrites, then  $(X' \cup X'', Y' \cup Y'')$  is a relative dendrite (by  $\check{Z}1$ ), which means that  $(X, Y)$  is a relative dendrite (Prop. 5 (1)).

iii) Suppose we have  $Z = Z' \cap Z''$ , with  $Z', Z'', Z' \cup Z'' \in \mathbb{D}$ . We set  $Z' = \alpha X' \ddot{\cup} Y'$  and  $Z'' = \alpha X'' \ddot{\cup} Y''$ . If  $(X', Y'), (X'', Y''), (X' \cup X'', Y' \cup Y'')$  are relative dendrites, then  $(X' \cap X'', Y' \cap Y'')$  is a relative dendrite (by  $\check{Z}2$ ), which means that  $(X, Y)$  is a relative dendrite (Prop. 5 (2)).

By i), ii), and iii), we may affirm, by induction on  $\langle \mathbb{C}; D1, D2 \rangle$ , that  $(X, Y)$  is a relative dendrite whenever  $\alpha X \ddot{\cup} Y$  is a dendrite. □

## 7 Dyads and Dendrites

The goal of this section is to derive a theorem (Th. 4) which makes clear the link between dyads and dendrites, this link is formulated with the notion of a relative dendrite. The proof of the theorem is made possible mainly thanks to the previous Th. 2 and 3 and the following propositions.

In the following proposition, and by the convention introduced in Section 2, the notation  $\alpha A$  implicitly means that  $\alpha$  is disjoint from the cell  $A$ . Thus, since  $X \preceq Y \preceq A$ ,  $\alpha X \cup Y$  is a cone on  $(X, Y)$  and  $\alpha Y_A^* \cup X_A^*$  is a cone on  $(Y_A^*, X_A^*)$ .

**Proposition 6.** *If  $A \in \mathbb{C}$ , and  $X \preceq Y \preceq A$ , then  $(\alpha X \ddot{\cup} Y)_{\alpha A}^* = \alpha Y_A^* \ddot{\cup} X_A^*$ .*

**Proof.** We have  $(\alpha X \cup Y)_{\alpha A}^* = (\alpha X)_{\alpha A}^* \cap Y_{\alpha A}^*$ . But  $(\alpha X)_{\alpha A}^* = \alpha X_A^*$  (by Cor. 1 of [14]) and  $Y_{\alpha A}^* = \alpha Y_A^* \cup A$  (by Cor. 2 of [14]). Thus,  $(\alpha X \cup Y)_{\alpha A}^* = \alpha X_A^* \cap (\alpha Y_A^* \cup A) = \alpha Y_A^* \cup X_A^*$  (since  $Y_A^* \preceq X_A^*$  and  $X_A^* \preceq A$ ). □

**Proposition 7.** *Let  $(X, Y)$  be a relative dendrite. We have  $X \in \mathbb{D}$  if and only if  $Y \in \mathbb{D}$ .*

**Proof.** Let  $(X, Y) \in \ddot{\mathbb{S}}$  such that  $\alpha X \ddot{\cup} Y$  is a dendrite (see Th. 3).

i) Suppose  $Y \in \mathbb{D}$ . Since  $\alpha X \in \mathbb{D}$  (Prop. 6 of [14]) then, by D2, we have  $\alpha X \cap Y = X \in \mathbb{D}$ .

ii) Let  $A \in \mathbb{C}$  such that  $Y \preceq A$ , we suppose that  $\alpha$  is disjoint from  $A$ . Suppose  $X \in \mathbb{D}$ . Thus,  $X_A^* \in \mathbb{D}$  and  $(\alpha X \cup Y)_{\alpha A}^* \in \mathbb{D}$  (Prop. 11 of [14]). But  $(\alpha X \cup Y)_{\alpha A}^* = \alpha Y_A^* \cup X_A^*$  (Prop. 6). Since  $\alpha Y_A^* \in \mathbb{D}$ , by D2, it implies that  $\alpha Y_A^* \cap X_A^* = Y_A^* \in \mathbb{D}$ , thus  $Y \in \mathbb{D}$  (Prop. 11 of [14]).  $\square$

**Lemma 1.** *The collection  $\langle \ddot{\mathbb{C}}^+; \ddot{\mathbb{Z}}1, \ddot{\mathbb{Z}}2 \rangle$  satisfies  $\langle \ddot{\mathbb{Y}}1 \rangle$  and  $\langle \ddot{\mathbb{Y}}2 \rangle$ .*

**Proof**

i) In  $\langle \ddot{\mathbb{Z}}1 \rangle$ , if we replace  $S$  by  $S \cap T$ ,  $S'$  by  $S$ , and  $T'$  by  $S$ , we obtain:

-> If  $(S \cap T, T), (S, S), (S \cap T, S \cap T) \in \ddot{\mathbb{K}}$ , then  $(S, S \cup T) \in \ddot{\mathbb{K}}$ .

ii) In  $\langle \ddot{\mathbb{Z}}2 \rangle$ , if we replace  $T$  by  $S \cup T$ ,  $S'$  by  $T$ , and  $T'$  by  $T$ , we obtain:

-> If  $(S, S \cup T), (T, T), (S \cup T, S \cup T) \in \ddot{\mathbb{K}}$ , then  $(S \cap T, T) \in \ddot{\mathbb{K}}$ .

iii) If  $X \in \mathbb{S}$ , then  $\alpha X \ddot{\cup} X = \alpha X$  is a dendrite. Thus, by Th. 3,  $(X, X)$  is a relative dendrite. In consequence, if  $\ddot{\mathbb{K}}$  is the collection made of all relative dendrites, we obtain  $\langle \ddot{\mathbb{Y}}1 \rangle$  and  $\langle \ddot{\mathbb{Y}}2 \rangle$  from i) and ii), respectively.  $\square$

The following is easy to check.

**Proposition 8.** *Let  $X, Y \in \mathbb{S}$ .*

1) *If  $X, Y \in \mathbb{D}$ , then  $X \cap Y \in \mathbb{D}$  if and only if  $X \cup Y \in \mathbb{D}$ .*

2) *If  $X, X \cap Y \in \mathbb{D}$ , then  $Y \in \mathbb{D}$  if and only if  $X \cup Y \in \mathbb{D}$ .*

3) *If  $X, X \cup Y \in \mathbb{D}$ , then  $Y \in \mathbb{D}$  if and only if  $X \cap Y \in \mathbb{D}$ .*

4) *If  $X \cap Y, X \cup Y \in \mathbb{D}$ , then  $X \in \mathbb{D}$  if and only if  $Y \in \mathbb{D}$ .*

**Lemma 2.** *The collection  $\langle \ddot{\mathbb{C}}^+; \ddot{\mathbb{Z}}1, \ddot{\mathbb{Z}}2 \rangle$  satisfies  $\langle \ddot{\mathbb{T}} \rangle$ ,  $\langle \ddot{\mathbb{U}} \rangle$ , and  $\langle \ddot{\mathbb{L}} \rangle$ .*

**Proof.** Let  $(R, S) \in \ddot{\mathbb{S}}$  and  $(S, T) \in \ddot{\mathbb{S}}$ , and let  $\alpha, \beta$  be two distinct vertices disjoint from  $T$ . Note that  $\alpha R \cup S \preceq \alpha R \cup T$ . We set  $U = \beta(\alpha R \ddot{\cup} S) \ddot{\cup} (\alpha R \ddot{\cup} T)$ .

i) We observe that  $U = (\alpha\beta R) \cup (\beta S \cup T)$ . We have  $\alpha\beta R \in \mathbb{D}$ , and  $(\alpha\beta R) \cap (\beta S \cup T) = \beta R \in \mathbb{D}$ . Thus, by Prop. 8 (2),  $U \in \mathbb{D}$  if and only if  $\beta S \cup T \in \mathbb{D}$ , i.e., if and only if  $(S, T)$  is a relative dendrite (Th. 3).

ii) Suppose  $(S, T)$  is a relative dendrite. By i) and Th. 3,  $(\alpha R \ddot{\cup} S, \alpha R \ddot{\cup} T)$  is a relative dendrite. By Prop. 7,  $\alpha R \ddot{\cup} S$  is a dendrite if and only if  $\alpha R \ddot{\cup} T$  is a dendrite. By Th. 3, it follows that  $(R, S)$  is a relative dendrite if and only if  $(R, T)$  is a relative dendrite. This fact allows us to affirm that the collection  $\langle \ddot{\mathbb{C}}^+; \ddot{\mathbb{Z}}1, \ddot{\mathbb{Z}}2 \rangle$  satisfies  $\langle \ddot{\mathbb{T}} \rangle$  and  $\langle \ddot{\mathbb{L}} \rangle$ .

iii) Suppose that  $(R, S)$  and  $(R, T)$  are relative dendrites, thus  $\alpha R \ddot{\cup} S$  and  $\alpha R \ddot{\cup} T$  are dendrites (Th. 3). We have  $U = \beta(\alpha R \ddot{\cup} S) \ddot{\cup} (\alpha R \ddot{\cup} T)$  and  $\beta(\alpha R \ddot{\cup} S) \cap (\alpha R \ddot{\cup} T) = \alpha R \ddot{\cup} S$ . Thus, we see that, by D1, the complex  $U$  is a dendrite. By i), it follows that  $(S, T)$  is a relative dendrite. This last fact allows us to affirm that the collection  $\langle \ddot{\mathbb{C}}^+; \ddot{\mathbb{Z}}1, \ddot{\mathbb{Z}}2 \rangle$  satisfies  $\langle \ddot{\mathbb{U}} \rangle$ .  $\square$

**Lemma 3.** *If  $X \in \mathbb{D}$ , then  $(\emptyset, X) \in \ddot{\mathbb{X}}$ .*

**Proof**

- i) If  $X \in \mathbb{C}$ , then  $(\emptyset, X) \in \check{\mathbb{C}}$ . Thus  $(\emptyset, X) \in \check{\mathbb{X}}$ .
- ii) Let  $S, T \in \mathbb{D}$  such that  $S \cap T \in \mathbb{D}$ . By (D1), we have  $S \cup T \in \mathbb{D}$ . Suppose  $(\emptyset, S), (\emptyset, T), (\emptyset, S \cap T) \in \check{\mathbb{X}}$ . We have  $(S \cap T, T) \in \check{\mathbb{X}}$  (Prop. 1 and  $\langle \check{U} \rangle$ ). Therefore  $(S, S \cup T) \in \check{\mathbb{X}}$  (Th. 2 and  $\langle \check{Y}1 \rangle$ ). Then  $(\emptyset, S \cup T) \in \check{\mathbb{X}}$  (Prop. 1 and  $\langle \check{T} \rangle$ ).
- iii) Let  $S, T \in \mathbb{D}$  such that  $S \cup T \in \mathbb{D}$ . By (D2), we have  $S \cap T \in \mathbb{D}$ . Suppose  $(\emptyset, S), (\emptyset, T), (\emptyset, S \cup T) \in \check{\mathbb{X}}$ . We have  $(S, S \cup T) \in \check{\mathbb{X}}$  (Prop. 1 and  $\langle \check{U} \rangle$ ). Therefore  $(S \cap T, T) \in \check{\mathbb{X}}$  (Th. 2 and  $\langle \check{Y}2 \rangle$ ). Then  $(\emptyset, S \cap T) \in \check{\mathbb{X}}$  (Prop. 1 and  $\langle \check{L} \rangle$ ).
- By the very definition of a dendrite, the result follows by induction.  $\square$

The following theorem is one of the main results of the paper. Intuitively, it asserts that, if  $(X, Y)$  is a dyad, then we cancel out all cycles of  $Y$  (*i.e.*, we obtain an acyclic complex), whenever we cancel out those of  $X$  (by the way of a cone, see Th. 3). Furthermore, Th. 4 asserts that, if we are able to cancel all cycles of  $Y$  by such a way, then  $(X, Y)$  is a dyad.

**Theorem 4.** *Let  $(X, Y) \in \check{\mathbb{S}}$ . We have  $(X, Y) \in \check{\mathbb{X}}$  if and only if  $(X, Y)$  is a relative dendrite.*

**Proof**

- i) Suppose  $(X, Y)$  is a relative dendrite, *i.e.*,  $(X, Y) \in \langle \check{\mathbb{C}}^+; \check{\mathbb{Z}}1, \check{\mathbb{Z}}2 \rangle$ . By Th. 3, we have  $\alpha X \check{U} Y \in \mathbb{D}$  and, by Lemma 3,  $(\emptyset, \alpha X \check{U} Y) \in \check{\mathbb{X}}$ . We also have  $(\emptyset, \alpha X) \in \check{\mathbb{X}}$  (Prop. 2). It means that  $(\alpha X, \alpha X \check{U} Y) \in \check{\mathbb{X}}$  (Prop. 1 and  $\langle \check{U} \rangle$ ). We obtain  $(\alpha X \cap Y, Y) = (X, Y) \in \check{\mathbb{X}}$  (Th. 2 and  $\langle \check{Y}2 \rangle$ ). Thus,  $\langle \check{\mathbb{C}}^+; \check{\mathbb{Z}}1, \check{\mathbb{Z}}2 \rangle \subseteq \check{\mathbb{X}}$ .
- ii) The collection  $\langle \check{\mathbb{C}}^+; \check{\mathbb{Z}}1, \check{\mathbb{Z}}2 \rangle$  contains  $\check{\mathbb{C}}$  and satisfies  $\langle \check{Y}1 \rangle, \langle \check{Y}2 \rangle, \langle \check{T} \rangle, \langle \check{U} \rangle$ , and  $\langle \check{L} \rangle$  (Lemmas 1 and 2). Thus  $\langle \check{\mathbb{C}}; \check{Y}1, \check{Y}2, \check{T}, \check{U}, \check{L} \rangle \subseteq \langle \check{\mathbb{C}}^+; \check{\mathbb{Z}}1, \check{\mathbb{Z}}2 \rangle$  (see remark 1). By Th. 2, the result is  $\check{\mathbb{X}} \subseteq \langle \check{\mathbb{C}}^+; \check{\mathbb{Z}}1, \check{\mathbb{Z}}2 \rangle$ .  $\square$

Trivially, we have  $X \in \mathbb{D}$  if and only if  $\alpha \emptyset \check{U} X \in \mathbb{D}$ . Thus, by Th. 3,  $X \in \mathbb{D}$  if and only if  $(\emptyset, X)$  is a relative dendrite. It follows that, as a direct consequence of Th. 4, we have the following.

**Corollary 1.** *Let  $X \in \check{\mathbb{S}}$ . We have  $X \in \mathbb{D}$  if and only if  $(\emptyset, X) \in \check{\mathbb{X}}$ .*

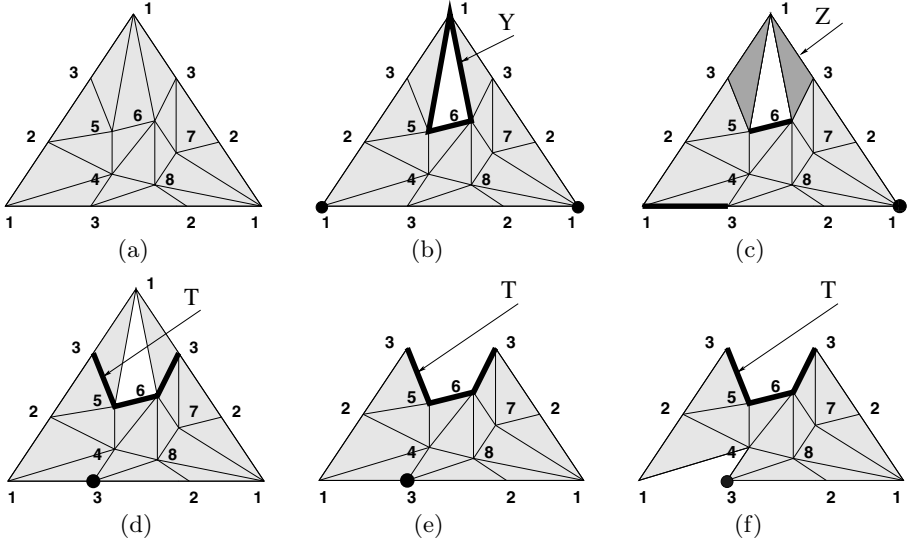
The following fact will be used for the illustration of the next section.

**Proposition 9.** *Let  $X, Y, Z \in \check{\mathbb{S}}$  such that  $X \preceq Y \preceq Z$ .*

*If  $Y$  collapses onto  $X$ , then  $(X, Z) \in \check{\mathbb{X}}$  if and only if  $(Y, Z) \in \check{\mathbb{X}}$ .*

*If  $Z$  collapses onto  $Y$ , then  $(X, Y) \in \check{\mathbb{X}}$  if and only if  $(X, Z) \in \check{\mathbb{X}}$ .*

**Proof.** If  $Y$  collapses onto  $X$ , then it may be seen that  $U' = \alpha Y \check{U} Z$  collapses onto  $V' = \alpha X \check{U} Z$ . Thus,  $U'$  is simple homotopic to  $V'$ . If  $Z$  collapses onto  $Y$ , then  $U'' = \alpha X \check{U} Z$  collapses onto  $V'' = \alpha X \check{U} Y$ . Again,  $U''$  is simple homotopic to  $V''$ . The result follows from Th. 3, Th. 4, and from Prop. 12 of [14]. This last proposition ensures that a complex  $S$  is a dendrite whenever it is simple homotopic to a dendrite.  $\square$



**Fig. 4.** (a): A triangulation  $D$  of the dunce hat, vertices with the same label have to be identified, (b): The complex  $X = D \setminus \{1, 5, 6\}$  and the complex  $Y \preceq X$  (highlighted), (c): The complex  $Z$  (highlighted) collapses onto  $Y$ , (d): The complex  $Z$  collapses onto  $T$  (highlighted), (e) and (f): The first steps of a collapse sequence of  $X$  onto  $T$ .

## 8 The Dunce Hat

We give, in this section, an illustration of the previous notions. We consider the complex  $D$  which is the triangulation of the dunce hat [16] depicted in Fig. 4 (a). As mentioned before, the dunce hat is contractible but not collapsible. In fact, it is possible to find a collapsible complex which collapses onto  $D$  (e.g., see Th.1 of [16]). This shows that  $D$  is a dendrite. In the following, we will see that it is possible to recognize  $D$  as a dendrite without considering any complex larger than  $D$  (by using only “internal moves”).

We consider the complex  $X = D \setminus \{1, 5, 6\}$ , we denote by  $C$  the cell whose facet is  $\{1, 5, 6\}$ , and by  $Y$  the complex  $Y = C \cap X$ , see Fig. 4 (b). We will see below that  $(Y, X) \in \check{\mathbb{X}}$ . By  $\langle \check{Y}1 \rangle$ , this fact implies  $(C, C \cup X) \in \check{\mathbb{X}}$ , i.e.,  $(C, D) \in \check{\mathbb{X}}$ . Since  $(\emptyset, C) \in \check{\mathbb{X}}$ , by  $\langle \check{T} \rangle$ , this implies  $(\emptyset, D) \in \check{\mathbb{X}}$ . Thus, by Cor. 1 of Th. 4, we get  $D \in \mathbb{D}$ . Now, we check that  $(Y, X) \in \check{\mathbb{X}}$  using Prop. 9:

- The complex  $Z$  of Fig. 4 (c) collapses onto  $Y$ , thus  $(Y, X) \in \check{\mathbb{X}}$  if  $(Z, X) \in \check{\mathbb{X}}$ ;
- $Z$  collapses onto the complex  $T$  of Fig. 4 (d), thus  $(Z, X) \in \check{\mathbb{X}}$  if  $(T, X) \in \check{\mathbb{X}}$ ;
- It could be checked that  $X$  collapses onto  $T$ , the first steps of a collapse sequence are given 4 (e) and (f). Thus, since  $(T, T) \in \check{\mathbb{X}}$ , we have  $(T, X) \in \check{\mathbb{X}}$ .

## 9 Conclusion

We introduced several axioms for describing dyads, i.e., pair of complexes which are, in a certain sense, linked by a “relative topology”. Our two main results are

theorems 3 and 4 which make clear the links between dyads and dendrites, *i.e.*, between dyads and acyclic complexes.

We proposed an approach which is exclusively based on discrete notions and also, by the means of completions, on constructions on sets.

In the future, we will further investigate the possibility to develop a discrete framework related to combinatorial topology.

## References

1. Whitehead, J.H.C.: Simplicial spaces, nuclei, and  $m$ -groups. Proc. London Math. Soc. (2) 45, 243–327 (1939)
2. Björner, A.: Topological methods. In: Graham, R., Grötschel, M., Lovász, L. (eds.) Handbook of Combinatorics, pp. 1819–1872. North-Holland, Amsterdam (1995)
3. Hachimori, M.: Nonconstructible simplicial balls and a way of testing constructibility. Discrete Comp. Geom. 22, 223–230 (1999)
4. Kahn, J., Saks, M., Sturtevant, D.: A topological approach to evasiveness. Combinatorica 4, 297–306 (1984)
5. Welker, V.: Constructions preserving evasiveness and collapsibility. Discrete Math. 207, 243–255 (1999)
6. Jonsson, J.: Simplicial Complexes of Graphs. Springer (2008)
7. Kalai, G.: Enumeration of  $Q$ -acyclic simplicial complexes. Israel Journal of Mathematics 45(4), 337–351 (1983)
8. 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)
9. 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)
10. Bertrand, G.: On critical kernels. Comptes Rendus de l'Académie des Sciences, Série Math. (345), 363–367 (2007)
11. Rosenfeld, A.: Digital topology. Amer. Math. Monthly, 621–630 (1979)
12. Kovalevsky, V.: Finite topology as applied to image analysis. Comp. Vision Graphics, and Im. Proc. 46, 141–161 (1989)
13. Kong, T.Y., Rosenfeld, A.: Digital topology: introduction and survey. Comp. Vision, Graphics and Image Proc. 48, 357–393 (1989)
14. Bertrand, G.: Completions and simplicial complexes, HAL-00761162 (2012)
15. Bing, R.H.: Some aspects of the topology of 3-manifolds related to the Poincaré Conjecture. In: Lectures on Modern Mathematics II, pp. 93–128. Wiley (1964)
16. Zeeman, E.C.: On the dunce hat. Topology 2, 341–358 (1964)
17. Serra, J.: Image Analysis and Mathematical Morphology, part II: theoretical advances. Academic Press, London (1988)
18. Aczel, P.: An introduction to inductive definitions. In: Barwise, J. (ed.) Handbook of Mathematical Logic, pp. 739–782 (1977)

# Asymptotic Analysis and Random Sampling of Digitally Convex Polyominoes

O. Bodini<sup>1</sup>, Ph. Duchon<sup>2</sup>, A. Jacquot<sup>1</sup>, and L. Mutafchiev<sup>3,4</sup>

<sup>1</sup> LIPN, Université Paris 13, 99, Avenue Jean-Baptiste Clément 93430 Villetaneuse, France

<sup>2</sup> LaBRI, 351, cours de la Libération F-33405 Talence cedex, France

<sup>3</sup> American University in Bulgaria 1 Georgi Izmirliiev Sq. Blagoevgrad 2700, Bulgaria,

<sup>4</sup> Institute of Mathematics and Informatics Acad. Georgi Bonchev Str., Block 8 1113 Sofia, Bulgaria

**Abstract.** Recent work of Brlek *et al.* gives a characterization of digitally convex polyominoes using combinatorics on words. From this work, we derive a combinatorial symbolic description of digitally convex polyominoes and use it to analyze their limit properties and build a uniform sampler. Experimentally, our sampler shows a limit shape for large digitally convex polyominoes.

## Introduction

In discrete geometry, a finite set of unit square cells is said to be a *digitally convex polyomino*<sup>1</sup> if it is exactly the set of unit cells included in a convex region of the plane. We only consider digitally convex polyominoes up to translation. The *perimeter* of a digitally convex polyomino is that of the smallest rectangular box that contains it.

Brlek *et al.* [7] described a characterization of digitally convex polyominoes, in terms of words coding their contour. In this paper, we reformulate this characterization in the context of constructible combinatorial classes and we use it to build and analyze an algorithm to randomly sample digitally convex polyominoes.

Our algorithm, based on a model of parametrized samplers, called Boltzmann samplers [8], draws digitally convex polyominoes at random. Although all possible digitally convex polyominoes have positive probability, the perimeter of the randomly generated polyomino being itself random, two different structures with the same perimeter appear with the same probability. Moreover, an appropriate choice of the tuning parameter allows the user to adjust the random model, typically in order to generate *large* structures. We present also in this paper how to tune this parameter.

---

<sup>1</sup> The usual definition of a polyomino requires the set to be connected, whereas digitally convex sets may be disconnected. However, one can coherently define some polygon as the boundary of any digitally convex polyomino; in the case of disconnected sets, this boundary will not be self-avoiding.

The Boltzmann model of random sampling, as introduced in [8], is a general method for the random generation of discrete combinatorial structures where, for some real parameter  $x > 0$ , each possible structure  $\gamma$ , with (integer) size  $|\gamma|$ , is obtained with probability proportional to  $x^{|\gamma|}$ . A *Boltzmann sampler* for a combinatorial class  $\mathcal{C}$  is a randomized algorithm that takes as input a parameter  $x$ , and outputs a random element of  $\mathcal{C}$  according to the Boltzmann distribution with parameter  $x$ .

Samples obtained via this generator suggested that large random quarters of digitally convex polyominoes exhibit a limit shape. We identify and prove this limit shape in Section 2

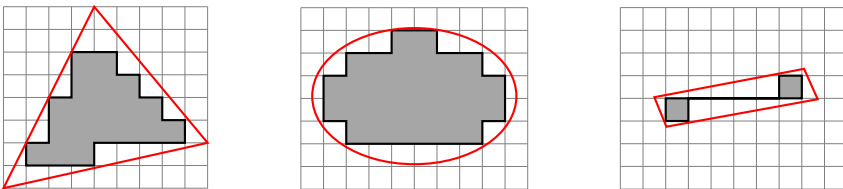
The first section is dedicated to introducing the characterization of Brlek *et al* [7] in the framework of symbolic methods. In Section 2, we analyze asymptotic properties of quarters of digitally convex polyominoes. Finally, we give in Section 3 the samplers for digitally convex polyominoes and some analysis for the complexity of the sampling. Due to the lack of space, all proofs are postponed.

## 1 Characterization of Digitally Convex Polygons

The goal of this section is to recall (without proofs) the characterization by Brlek, Lachaud, Provençal, Reutenauer [7] of digitally convex polyominoes and recast it in terms of the symbolic method. This characterization is the starting point to efficiently sample large digitally convex polyominoes, and is thus needed in the next chapters.

### 1.1 Digitally Convex Polyominoes

**Definition 1.** A digitally convex polyomino, *DCP* for short, is the set of all cells of  $\mathbb{Z}^2$  included in a bounded convex region of the plane.



**Fig. 1.** A few digitally convex polyominoes (in grey) of perimeter 24, 26 and 16, and their contour (in black)

A first geometrical characterization directly follows from the definition: a set of cells of the square lattice  $P$  is a digitally convex polyomino if all cells included in the convex hull of  $P$  are in  $P$ .

For our propose, a DCP will be rather characterized through its *contour*.



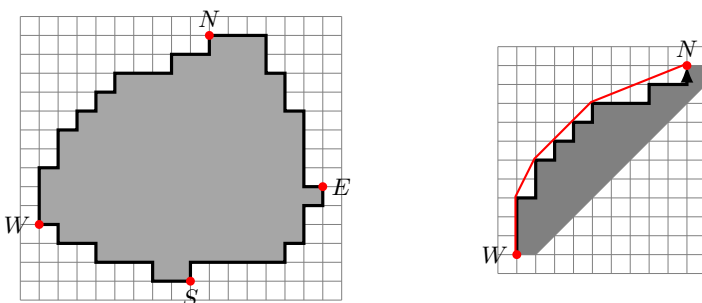
**Definition 2.** *The contour of a polyomino  $P$  is the closed non-crossing path on the square lattice with no half turn allowed such that its interior is  $P$ . In the case where  $P$  is not connected, we take as the contour, the only such path which stays inside the convex hull of  $P$ .*

We define the *perimeter* of  $P$  to be the length of the contour (note that, for digitally convex polyominoes, it is equal to the perimeter of the smallest rectangular box that contains  $P$ ).

The contour of DCP can be decomposed into four specifiable sub-paths through the standard decomposition of polyominoes.

The standard decomposition of a polyomino distinguishes four extremal points:

- $W$  is the lowest point on the leftmost side
- $N$  is the leftmost point in the top side
- $E$  is the highest point on the rightmost side
- $S$  is the rightmost point on the bottom side



**Fig. 2.** A polyomino composed of 4 NW-convex. The  $W$  to  $N$  NW-convex path is coded by  $w = 11101101010100010001$ .

The contour of a DCP is then the union of the four (clockwise) paths  $WN$ ,  $NE$ ,  $ES$  and  $SW$ . Rotating the latter three paths by, respectively, a quarter turn, a half turn, three quarter turn counterclockwise leaves all paths containing only north and east steps; digital convexity is characterized by the fact that each (rotated) side is *NW-convex*.

**Definition 3.** *A path  $p$  is NW-convex if it begins and ends by a north step and if there is no cell between  $p$  and its upper convex hull (see Fig. 2).*

In the following, we mainly focus on the characterization and random sampling of NW-convex paths.

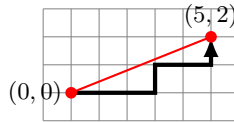
### 1.2 Words

The characterization in [7] is based on combinatorics on words. Let us recall some classical notations. We are interested in words on the alphabet  $\{0, 1\}$ . Thus  $\{0, 1\}^*$  is the set of all words,  $\{0, 1\}^+$  is the set of non-empty words.

Each NW-path can be bijectively coded by a word  $w \in \{0, 1\}^*$ . From  $W$  to  $N$ , the letter 0 encodes a horizontal (east) step and 1 a vertical (north) step (see figure 2).

The idea is to decompose a NW-convex path  $w$  by contacts between  $w$  and its convex hull.

**Definition 4.** Let  $p, q$  be two integers, with  $p \geq 0$  and  $q > 0$ . The Christoffel word associated to  $p, q$  is the word which codes the highest path going from  $(0, 0)$  to  $(p, q)$  while staying under the line going from  $(0, 0)$  to  $(p, q)$ . A Christoffel word is primitive if  $p$  and  $q$  are coprime.



**Fig. 3.** The Christoffel primitive word 0001001, of slope  $2/5$

Note that a Christoffel primitive word always ends with 1.

### 1.3 Symbolic Characterization of NW-Convex Paths

Let us recall in this section, two basic notions in analytic combinatorics: the combinatorial classes and the enumerative generating functions.

A *combinatorial class* is a finite or countable set  $\mathcal{C}$ , together with a *size function*  $|\cdot| : \mathcal{C} \mapsto \mathbb{N}$  such that, for all  $n \in \mathbb{N}$ , only a finite number  $c_n$  of elements of  $\mathcal{C}$  have size  $n$ . The (ordinary) *generating function*  $C(z)$  for the class  $\mathcal{C}$  is the formal power series  $C(z) = \sum_n c_n z^n = \sum_{\gamma \in \mathcal{C}} z^{|\gamma|}$ . If  $C(z)$  has a positive (possibly infinite) convergence radius  $\rho$  (which is equivalent to the condition that  $\overline{\lim} c_n^{1/n} < \infty$ ), standard theorems in analysis imply that the power series  $C(z)$  converges and defines an analytic function in the complex domain  $|z| < \rho$ ; here we will only use the fact that  $C(z)$  is defined for real  $0 < z < \rho$ .

Our sampler is based on the following decomposition theorem.

**Theorem 1.** [7] *A word is NW-convex if and only if it is a sequence of Christoffel primitive words of decreasing slope, beginning with 1.*

The reason of a NW-convex path to begin with a vertical step is to avoid half-turn on the contour of a polyomino. Indeed, since all Christoffel primitive words end with 1, this ensures compatibility with the standard decomposition (beginning and ending on a corner). Since the Christoffel primitive words appear in decreasing order, NW-convex paths can be identified with multisets of Christoffel primitive words, with the condition that the word 1 appears at least once in the multiset (this condition can be removed by removing the initial vertical step from NW-convex paths). This is the description we will use in what follows next.

The generating function of Christoffel primitive words, counted by their lengths, is  $\sum_{n \geq 1} \varphi(n)z^n$ , with  $\varphi$  the Euler's totient function. It follows [11, p. 29] that the generating function of the class  $\mathcal{S}$  of NW-convex paths, counted by length, is  $S(z) = \prod_{n=1}^{\infty} (1 - z^n)^{-\varphi(n)}$ . More precisely, we can use the 3-variate generating function  $S(z, h, v) = (1 - zv)^{-1} \prod_{n=2}^{\infty} \prod_{p+q=n, p \wedge q=1} (1 - z^n v^p h^q)^{-1}$ , to describe by  $[z^n h^i v^j]S(z, h, v)$  the number of NW-convex paths beginning in  $(0, 0)$  and terminating in position  $(i, j)$ .

## 2 Asymptotics for NW-Convex Paths and Its Limit Shape

This section is dedicated to the analysis of some properties of NW-convex paths. The main objective is to describe a limit shape for the normalized random NW-convex paths. This is obtained in three steps. In the first one we extract the asymptotic of NW-convex paths using a Mellin transform approach. In the second one, using the same approach we prove that the asymptotic of the average number of initial vertical steps of a NW-convex path is in  $O(\sqrt[3]{n})$ . Then, using some technical lemmas, we conclude with the fact that the limit shape is  $\sqrt{2z - z}$  with  $0 \leq z \leq 1/2$ . For more details, see [11].

The first step is to determine the Mellin transform of  $\ln(S(e^{-t}))$  which is easier than to obtain that the Mellin transform of  $S(z)$ . After that it is quite easy to compute the expansion for  $S(z)$  when  $z$  tends to 1.

**Lemma 1.** *The Mellin transform associated with the series of irreducible discrete segments is*

$$\mathcal{M}[\ln(S(e^{-t}))](s) = \frac{\zeta(s+1)\zeta(s-1)\Gamma(s)}{\zeta(s)},$$

where  $\zeta(z)$  and  $\Gamma(z)$  denote the Riemann zeta function and the Gamma function, respectively.

Using technical approach [10] following Mellin transform properties and Hayman's method, we can extract from the lemma 1, the asymptotic growth of the number of NW-convex paths of size  $n$ :

**Proposition 1.** *For the number  $p_{NW}(n)$  of NW-convex paths of size  $n$ , we have*

$$p_{NW}(n) \sim \alpha n^{-11/18} \exp\left(\beta n^{2/3} + g\left(\left(\frac{12\zeta(3)}{n\pi^2}\right)^{1/3}\right)\right),$$

with  $g(t) = \sum_r t^{-r} \Gamma(r) \zeta(r+1) \zeta(r-1)$  where  $r$  runs over the non-trivial zeros of the Riemann zeta function and

$$\alpha = \frac{1}{6} \frac{2^{5/9} e^{\frac{(5/2)\zeta(3) - 2\zeta'(-1)\pi^2}{\pi^2}} \sqrt[9]{\zeta(3)} 3^{\frac{11}{18}}}{\pi^{\frac{8}{9}}} \sim 0.3338488807\dots$$

$$\text{and } \beta = \frac{3}{2^{2/3}} \left( \frac{\zeta(3)}{\zeta(2)} \right)^{1/3} = \frac{2^{-1/3} 3^{4/3} \zeta(3)^{1/3}}{\pi^{2/3}} \sim 1.702263426\dots$$

*Remark 1.* The contribution of  $g\left(\left(\frac{12\zeta(3)}{n\pi^2}\right)^{1/3}\right)$  is a fluctuation of very small amplitude. In particular, this contribution is imperceptible on the first 1000 coefficients.

Now, we focus on the study of the average number of initial vertical steps (which corresponds to the size of the first block of 1 in its associated word) in a NW-convex path.

**Lemma 2.** *The average number of initial steps is equivalent to  $\frac{\sqrt[3]{18\pi^2 n}}{6\sqrt[3]{\zeta(3)}}$ .*

In particular, if we renormalize the NW-convex path by  $1/n$ , the contribution of the initial steps for the limit shape is null.

Now, we are interested in the average position of the terminating point of a random NW-convex path. If we consider NW-convex path without their initial vertical steps, then by symmetry, we can conclude that the average ending position is  $(x_n \sim \frac{n}{2}, y_n \sim \frac{n}{2})$ . But by lemma 2 and the fact that the length of the renormalized initial vertical steps is  $o(1)$ , it follows that:

**Lemma 3.** *The average position of the ending point of a random NW-convex path of size  $n$  is  $(x_n \sim \frac{n}{2}, y_n \sim \frac{n}{2})$ .*

Following the same approach, with a little more work, we can prove that:

**Proposition 2.** *The average abscissa of the point of slope  $x$  in a renormalized by  $1/n$  NW-convex path of size  $n$  is  $\frac{1}{2} \left( 1 - \left( \frac{x}{1+x} \right)^2 \right)$ .*

At this stage, to prove that the limit shape is deterministic, we need to show that the standard deviation of the abscissa is in  $o(n)$ . This proof is long and technical, but follows the same way that we do for the expectation. We conclude by solving the differential equation:  $\left\{ f(0) = 0, 2f(z) = 1 - \frac{\left(\frac{d}{dz}f(z)\right)^2}{\left(1 + \frac{d}{dz}f(z)\right)^2} \right\}$  which explains the fact that the slope of  $f(z)$  is  $x$  at the abscissa  $\frac{1}{2} \left( 1 - \left( \frac{x}{1+x} \right)^2 \right)$ .

Consequently, we have:

**Theorem 2.** *The limit shape for the renormalized by  $1/n$  NW-convex path of size  $n$  as  $n$  tends to the infinity is the curve of equation  $f(z) = \sqrt{2z - z}$ .*

### 3 Boltzmann Samplers for NW-Convex Paths

Boltzmann samplers have been introduced in 2004 by Duchon *et al.* [8] as a general framework to generate uniformly at random specifiable combinatorial structures. These samplers are very efficient, their expected complexity is typically linear. In comparison with the recursive method of sampling, the principle of Boltzmann samplers essentially deals with evaluations of the generating function of the structures, and avoids counting coefficient (which need to be pre-computed in the recursive method). Quite a few papers have been written to extend and optimize Boltzmann samplers [13,2,5,4,6,1,3].

Consequently, we choose Boltzmann sampling for the class of digitally convex polyominoes. After a short introduction to the method, we analyze the complexity of the sampler. This part is based on an analysis by Mellin transform techniques.

#### 3.1 A Short Introduction to Boltzmann Samplers

Let us recall the definitions and the main ideas of Boltzmann sampling.

**Definition 5.** *Let  $\mathcal{A}$  be a combinatorial class and  $A(x)$  its ordinary generating function. A (free) Boltzmann sampler with parameter  $x$  for the class  $\mathcal{A}$  is a random algorithm  $\Gamma_x \mathcal{A}$  that draws each object  $\gamma \in \mathcal{A}$  with probability:*

$$\mathbb{P}_x(\gamma) = \frac{x^{|\gamma|}}{A(x)}.$$

*Notice that this definition is consistent only if  $x$  is a positive real number taken within the disk of convergence of the series  $A(x)$ .*

The great advantage of choosing the Boltzmann distribution for the output size is to allow simple and automatic rules to design Boltzmann samplers from a specification of the class.

Note that from free Boltzmann samplers, we easily define two variants: the *exact-size Boltzmann sampler* and the *approximate-size* one, just by rejecting the inappropriate outputs until we obtain respectively a targeted size or a targeted interval of type  $[(1 - \varepsilon)n, (1 + \varepsilon)n]$ . In order to optimize this rejection phase, it is crucial to tune efficiently the parameter  $x$ . A good choice is generally to take the unique positive real solution  $x_n$  (or an approximation of it when  $n$  tends to infinity) of the equation  $\frac{x A'(x)}{A(x)} = n$  which means that the expected size of the output tuned by  $x_n$  equals  $n$ .

To conclude, let us recall that authors of the seminal paper [8] distinguished a special case where Boltzmann samplers are particularly efficient (and we prove in the sequel, that we are in this situation). This case arises when the Boltzmann distribution of the output is *bumpy*, that is to say, when the following conditions are satisfied:

- $\mu_1(x) \rightarrow \infty$  when  $x \rightarrow \rho^-$
- $\sigma(x)/\mu_1(x) \rightarrow 0$  when  $x \rightarrow \rho^-$ ,

where  $\mu_1(x)$  (resp.  $\sigma^2(x)$ ) is the expected size (resp. the variance) of the output, and  $\rho$  is the dominant singularity of  $A(x)$ .

### 3.2 The Class of the Digitally Convex Polyominoes

Let us recall that the digitally convex polyominoes can be decomposed into four NW-convex paths, each of them being a multiset of discrete irreducible segments. Moreover, according to the previous section, we have a specification for the discrete irreducible segments in terms of word theory. This brings us the generating function associated to a NW-convex path:

$$S(z) = \prod_{n=1}^{\infty} (1 - z^n)^{-\varphi(n)},$$

where  $\varphi(n)$  designs the Euler’s totient function.

The first question that occurs concerns the determination of the parameter  $x_n$  which is a central point for the tuning of the sampler. In order to approximate  $x_n$  as  $n$  tends to infinity, we need to apply the asymptotic of  $S(z)$  as  $z \rightarrow 1$ , we already calculate for the asymptotic of the NW-convex paths.

### 3.3 The Boltzmann Distribution of the NW-Convex Paths

The first step to analyze the complexity of exact- and approximate-size Boltzmann sampler is to characterize the type of the Boltzmann distribution. In this subsection we prove that the Boltzmann distribution is bumpy. This ensures that we only need on average a constant number of trials to draw an object of approximate-size. Moreover, a precise analysis allows us to give the complexity of the exact-size sampling.

Firstly, we derive from the equivalence of  $S(z)$  close to its dominant singularity  $\rho = 1$ , an expression for the tuning of the Boltzmann parameter:

**Corollary 1.** *A good choice for the Boltzmann parameter  $x_n$  in order to draw a large NW-convex paths of size  $n$  is  $x_n = 1 - \sqrt[3]{\frac{12\zeta(3)}{n\pi^2}}$ .*

So, the first condition for the bumpy distribution is clearly verified. We now focus our attention on the fraction  $\sigma(x)/\mu_1(x)$ .

**Lemma 4.** *The expected size of the Boltzmann output satisfies:*

$$\mu_1(x) \sim \frac{12\zeta(3)}{(1-x)^3 \pi^2} \text{ as } x \text{ tends to } 1.$$

*The variance of the size of the Boltzmann output satisfies:*

$$\sigma(x) \sim \frac{6 \sqrt{\zeta(3)x}}{\pi(1-x)^2} \text{ as } x \text{ tends to } 1.$$

*So, the Boltzmann distribution of the NW-convex paths is bumpy.*

Finally, we describe the sampler for digitally convex polyominoes. This needs some care during the stage when we glue the NW-convex paths together.

### 3.4 Random Sampler to Draw Christoffel Primitive Words

We now look more precisely how to implement the samplers. Firstly, we address the question of drawing two coprime integers which is non-classical in Boltzmann sampling, from which we derive a Boltzmann sampler for NW-convex paths. The first step to generate NW-convex paths is to draw Christoffel primitive words with Boltzmann probability. We recall that this is equivalent to draw two coprime integers  $p, q$  with probability  $\frac{x^{p+q}}{\sum_{n \geq 1} \varphi(n)x^n}$ .

Let  $b(x, n) := \frac{\varphi(n)x^n}{\sum_{n=1}^{\infty} \varphi(n)x^n}$ . The following algorithm is an elementary way to answer the question posed above:

---

#### Algorithm 1. $\Gamma_{CP}$

---

**Input:** a parameter  $x$

**Output:** Two coprime integers

- 1 Draw  $n$  with Boltzmann probability  $b(x, n)$
  - 2 **Do** Draw  $p$  uniformly in  $\{1, \dots, n\}$
  - 3 **While**  $p, n$  not coprime
  - 4 **return**  $(p, n - p)$
- 

The average complexity of the algorithm is in  $O(n \ln \ln(n))$ .

### 3.5 Random Sampler Drawing a NW-Convex Path

To draw a NW-convex path, we use the isomorphism between NW-convex paths and multisets of Christoffel primitive words. The multiset is a classical constructor, for which its Boltzmann sampler in the unlabelled case had been given in [9]. The idea is to draw with an appropriate distribution (called IndexMax distribution) an integer  $M$ , and then draw a random number of Christoffel primitive words with a Boltzmann sampler of parameter  $x^i$  and to replicate each drawn object  $i$  times, for all  $1 \leq i \leq M$ . Well chosen probabilities ensures the Boltzmann distribution.

Once we get a multiset of pairs of coprime integers, we can transform it into a NW-convex path coded on  $\{0, 1\}$  as follows:

- Draw a multiset  $m$  in  $\text{MSET}(PC)$ ,
- Sort the elements  $(p, q)$  of  $m$  in decreasing order according to  $q/p$ ,
- Transform each element into the discrete line of slope  $q/p$  coded on  $\{0, 1\}$ ,
- add a 1 at the beginning.

Clearly, this transformation has a complexity in  $O(n \ln(n))$ , due to the sorting.

### 3.6 Complexity of Sampling a NW-Convex Path in Approximate and Exact Size

The previous sections bring all needed elements to determine the complexity of the Boltzmann sampler for a NW-convex path. The two following theorems respectively tackle the complexity of the sampling in the case of approximate-size output or exact size output.

**Theorem 3.** *An approximate-size Boltzmann sampler for NW-convex paths succeeds in one trial with probability tending to 1 as  $n$  tends to infinity. The overall cost of approximate-size sampling is  $O(n \ln(n))$  on average.*

**Theorem 4.** *An exact-size Boltzmann sampler for NW-convex paths succeeds in mean number of trials tending to  $\kappa \cdot n^{2/3}$  with  $\kappa = \frac{\sqrt[9]{2} \sqrt[3]{3} \pi^{5/6}}{\sqrt[6]{\zeta(3)}} \approx 4.075517917\dots$  as  $n$  tends to infinity. The overall cost of exact-size sampling is  $O(n^{5/3} \ln(\ln(n)))$  on average.*

*Remark 2.* Since  $\varphi(n)$  grows slowly, the parameter  $x_n$  tuned to draw large objects will be close to 1, which gives big replication orders. A consequence is that we do not need to calculate the generating function of primitive Christoffel words to a huge order to have a good approximation of our probabilities.

### 3.7 Random Sampler Drawing Digitally Convex Polyominoes

We can now sample independent NW-convex paths with Boltzmann probability. We want to obtain an entire polyomino by gluing four (rotated) NW-convex path.

However, gluing a 4-tuple of NW-convex paths, we do not necessarily obtain the contour of a polyomino. Indeed, we need the following extra conditions: the four NW-convex paths should be non-crossing, and they need to form a closed walk with no half-turn.

---

#### Algorithm 2. $\Gamma_P$

---

**Input:** a parameter  $x$

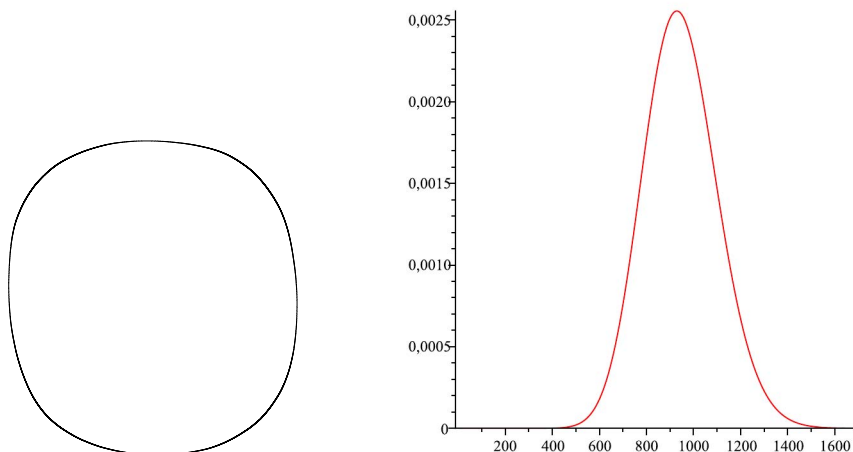
**Output:** a quadruple of compatible NW-convex paths.

- 1 **Repeat:**
  - 2     Draw WN, NE, ES, SW using independent calls to a Boltzmann sampler of
  - 3     NW-convex path of parameter  $x$ .
  - 4     **If**  $|WN|_0 + |NE|_1 = |ES|_0 + |SW|_1$  and  $|NE|_0 + |ES|_1 = |SW|_0 + |WN|_1$
  - 5     **then return**  $(WN, NE, ES, SW)$
- 

The non-crossing and no half-turn conditions are trivially satisfied when the four paths are NW-convex. Then the closing problem stays and we need to add a rejection phase at this step. More precisely, to be closed, we need to have as



much horizontal steps in the upper part from W to E as in the lower part from E to W, and as much vertical steps in the left part from S to N as in the right part from N to S. A naive way to sample DCP according to a Boltzmann distribution is presented in Algorithm 2. A more efficient uniform sampler can probably be adapted from [1] and is currently under development.



**Fig. 4.** To the left, a random polyomino of perimeter 81109 drawn with parameter  $x = 0.98$ . To the right perimeter distribution of a NW-convex path drawn with  $x = 0.8908086616$ .

## Conclusion

We proposed in this paper an effective way to draw uniformly at random digitally convex polyominoes. Our approach is based on Boltzmann generators which allows us to build large digitally convex polyominoes. These samples point out the fact that random digitally convex polyominoes admit a limit shape as their size tends to infinity. The limit shape of the NW-convex paths we proved in this paper seems to be also the limit shape for NW part of the digitally convex polyominoes. The tools to tackle it are for the moment beyond our reach. Even, the simpler question of a precise asymptotic enumeration of the digitally convex polyominoes (the order of magnitude is proven [12]) is currently a challenge. We conclude by noting that our work could certainly be extended to higher dimensions. But, this is a work ahead...

**Acknowledgement.** The fourth author established the collaboration on this work during his visit at LIPN, University of Paris 13. He wishes to thank for the fruitful discussions, hospitality and support. We thank Axel Bacher for its English corrections. Olivier Bodini is supported by ANR project MAGNUM, ANR 2010 BLAN 0204.

## References

1. Bodini, O., Gardy, D., Roussel, O.: Boys-and-girls birthdays and hadamard products. *Fundam. Inform.* 117(1-4), 85–101 (2012)
2. Bodini, O., Jacquot, A.: Boltzmann Samplers for Colored Combinatorial Objects. In: *Proceedings of Gascom 2008* (2008)
3. Bodini, O., Lumbroso, J.: Dirichlet random samplers for multiplicative combinatorial structures. In: *2012 Proceedings of the Eighth Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*, Kyoto, Japan, pp. 92–106 (2012)
4. Bodini, O., Ponty, Y.: Multi-dimensional boltzmann sampling of languages. *DMTCS Proceedings* (01), 49–64 (2010)
5. Bodini, O., Roussel, O., Soria, M.: Boltzmann samplers for first-order differential specifications. *Discrete Applied Mathematics* 160(18), 2563–2572 (2012)
6. Bodirsky, M., Fusy, E., Kang, M., Vigerske, S.: An unbiased pointing operator for unlabeled structures, with applications to counting and sampling. In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007*, pp. 356–365. Society for Industrial and Applied Mathematics, Philadelphia (2007)
7. Brlek, S., Lachaud, J.-O., Provencal, X., Reutenauer, C.: Lyndon christoffel digitally convex. *Pattern Recognition* 42(10), 2239–2246 (2009)
8. Duchon, P., Flajolet, P., Louchard, G., Schaeffer, G.: Boltzmann samplers for the random generation of combinatorial structures. *Combinatorics, Probability, and Computing* 13(4-5), 577–625 (2004); Special issue on Analysis of Algorithms
9. Flajolet, P., Fusy, E., Pivoteau, C.: Boltzmann sampling of unlabelled structures. In: *SIAM Press (ed.) Proceedings of ANALCO 2007 (Analytic Combinatorics and Algorithms) Conference*, vol. 126, pp. 201–211 (2007)
10. Flajolet, P., Salvy, B., Zimmermann, P.: Automatic average-case analysis of algorithm. *Theoretical Computer Science* 79(1), 37–109 (1991)
11. Flajolet, P., Sedgewick, R.: *Analytic Combinatorics*. Cambridge University Press (2009)
12. Ivic, A., Koplowitz, J., Zunic, J.D.: On the number of digital convex polygons inscribed into an  $(m, m)$ -grid. *IEEE Transactions on Information Theory* 40(5), 1681–1686 (1994)
13. Pivoteau, C., Salvy, B., Soria, M.: Boltzmann oracle for combinatorial systems. In: *Proceedings of the Fifth Colloquium on Mathematics and Computer Science: Algorithms, Trees, Combinatorics and Probabilities*, Blaubeuren, Germany, September 22-26. *Discrete Mathematics and Theoretical Computer Science*, pp. 475–488 (2008)

# Critical Connectedness of Thin Arithmetical Discrete Planes

Valérie Berthé<sup>1</sup>, Damien Jamet<sup>2</sup>, Timo Jolivet<sup>1,3</sup>, and Xavier Provençal<sup>4</sup>

<sup>1</sup> LIAFA CNRS, Université Paris Diderot, France

<sup>2</sup> LORIA, Université de Lorraine, France

<sup>3</sup> FUNDIM, Department of Mathematics, University of Turku, Finland

<sup>4</sup> LAMA, Université de Savoie, France

**Abstract.** The critical thickness of an arithmetical discrete plane refers to the infimum thickness that preserves its 2-connectedness. This infimum thickness can be computed thanks to a multidimensional continued fraction algorithm, namely the fully subtractive algorithm. We provide a characterization of the discrete planes with critical thickness that contain the origin and that are 2-connected.

## 1 Introduction

This paper studies the connectedness of thin arithmetic discrete planes in the three-dimensional space. We focus on the notion of 2-connectedness, and we restrict ourselves to planes with zero intercept that have critical thickness, that is, planes whose thickness is the infimum of the set of all the  $\omega \in \mathbb{R}_+$  such that the plane of thickness  $\omega$  is 2-connected (see Definitions 2.1 and 2.5). Let us recall that standard arithmetic discrete planes are known to be 2-connected, whereas naive ones are too thin to be 2-connected. We thus consider planes with a thickness that lies between the naive and the standard cases.

The problem of the computation of the critical thickness was raised in [6]. It has been answered in [9,10,7], with an algorithm that can be expressed in terms of a multidimensional continued fraction algorithm, namely the so-called fully subtractive algorithm. This algorithm explicitly yields the value of the critical thickness when it halts, and this value can be computed when the algorithm enters a loop (possibly infinite). Furthermore, the set  $\mathcal{F}_3$  of vectors for which the algorithm enters an infinite loop has Lebesgue measure zero, as a consequence of results of [12] in the context of a percolation model defined by rotations on the unit circle. Our main result is that a discrete plane with intercept zero and critical thickness is 2-connected when its normal vector belongs to  $\mathcal{F}_3$ . We also prove that vectors in  $\mathcal{F}_3$  are the only ones for which critical arithmetical discrete planes are 2-connected.

Our methods rely on a combinatorial generation method based on the notion of substitution for the planes under study (see Section 2.3 for more details). In Section 3 we construct a sequence of finite patterns  $(\mathbf{T}_n)_n$  of the planes with critical thickness, and we prove that these patterns are all 2-connected when

the parameters belong to  $\mathcal{F}_3$ . We then relate these finite patterns with thinner patterns  $\mathbf{P}_n$  that belong to the naive discrete plane with same parameters. These pattern are generated in terms of a geometric interpretation of the fully subtractive algorithm. Next, the thinner patterns  $\mathbf{P}_n$  are proved in Section 4 to generate the full naive plane. This yields the 2-connectedness of the critical plane (see Section 5). In other words, we use the fact that the underlying naive plane provides a relatively dense skeleton of the critical plane.

## 2 Basic Notions and Notation

### 2.1 Discrete and Stepped Planes

Let  $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$  be the canonical basis of  $\mathbb{R}^3$ , and let  $\langle \cdot, \cdot \rangle$  stand for the usual scalar product on  $\mathbb{R}^3$ . Given  $\mathbf{v} \in \mathbb{R}^3$  and  $i \in \{1, 2, 3\}$ , we let  $\mathbf{v}_i = \langle \mathbf{v}, \mathbf{e}_i \rangle$  denote the  $i^{\text{th}}$ -coordinate of  $\mathbf{v}$  in the basis  $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ .

**Definition 2.1 (Arithmetical discrete plane [13,1]).** *Given  $\mathbf{v} \in \mathbb{R}_+^3$ ,  $\mu \in \mathbb{R}$  and  $\omega \in \mathbb{R}_+$ , the arithmetical discrete plane with normal vector  $\mathbf{v}$ , intercept  $\mu$ , and thickness  $\omega$  is the set  $\mathfrak{P}(\mathbf{v}, \mu, \omega)$  defined as follows:*

$$\mathfrak{P}(\mathbf{v}, \mu, \omega) = \{\mathbf{x} \in \mathbb{Z}^3 : 0 \leq \langle \mathbf{x}, \mathbf{v} \rangle + \mu < \omega\}.$$

If  $\omega = \|\mathbf{v}\|_\infty = \max\{|\mathbf{v}_1|, |\mathbf{v}_2|, |\mathbf{v}_3|\}$  (resp.  $\omega = \|\mathbf{v}\|_1 = |\mathbf{v}_1| + |\mathbf{v}_2| + |\mathbf{v}_3|$ ), then  $\mathfrak{P}(\mathbf{v}, \mu, \omega)$  is said to be a naive arithmetical discrete plane (resp. standard arithmetical discrete plane).

Even if any finite subset of a digitized plane can be represented as a subset of an arithmetical discrete plane with integer parameters, we do not restrict ourselves here with finite sets, and we consider general arithmetical discrete with possibly non-integer parameters.

We will also deal with another discrete approximation of Euclidean planes, namely *stepped planes*. They can be considered as a more geometrical version, in the sense that they consist of *unit faces* instead of just points of  $\mathbb{Z}^3$ .

**Definition 2.2 (Unit faces, stepped planes).** *A unit face  $[\mathbf{x}, i]^*$  is defined as:*

$$\begin{aligned} [\mathbf{x}, 1]^* &= \{\mathbf{x} + \lambda \mathbf{e}_2 + \mu \mathbf{e}_3 : \lambda, \mu \in [0, 1]\} = \begin{array}{c} \blacktriangleleft \\ \blacktriangleleft \end{array} \\ [\mathbf{x}, 2]^* &= \{\mathbf{x} + \lambda \mathbf{e}_1 + \mu \mathbf{e}_3 : \lambda, \mu \in [0, 1]\} = \begin{array}{c} \blacktriangleleft \\ \bullet \\ \blacktriangleleft \end{array} \\ [\mathbf{x}, 3]^* &= \{\mathbf{x} + \lambda \mathbf{e}_1 + \mu \mathbf{e}_2 : \lambda, \mu \in [0, 1]\} = \begin{array}{c} \blacktriangleleft \\ \bullet \\ \blacktriangleleft \end{array} \end{aligned}$$

where  $i \in \{1, 2, 3\}$  is the type of  $[\mathbf{x}, i]^*$ , and  $\mathbf{x} \in \mathbb{Z}^3$  is the distinguished vertex of  $[\mathbf{x}, i]^*$ . Let  $\mathbf{v} \in \mathbb{R}_+^3$ . The stepped plane  $\Gamma_{\mathbf{v}}$  is the union of unit faces defined by:

$$\Gamma_{\mathbf{v}} = \{[\mathbf{x}, i]^* : 0 \leq \langle \mathbf{x}, \mathbf{v} \rangle < \langle \mathbf{e}_i, \mathbf{v} \rangle\}.$$

The notation  $\mathbf{x} + [\mathbf{y}, i]^*$  stands for the unit face  $[\mathbf{x} + \mathbf{y}, i]^*$ .

*Remark 2.3.* The set of distinguished vertices of  $\Gamma_{\mathbf{v}}$  is the naive arithmetical discrete plane  $\mathfrak{P}(\mathbf{v}, 0, \|\mathbf{v}\|_\infty)$ , whereas the set of all vertices of the faces of  $\Gamma_{\mathbf{v}}$  is the standard arithmetical discrete plane  $\mathfrak{P}(\mathbf{v}, 0, \|\mathbf{v}\|_1)$ .

## 2.2 Connecting Thickness and the Fully Subtractive Algorithm

**Definition 2.4 (Adjacency, connectedness).** *Two distinct elements  $\mathbf{x}$  and  $\mathbf{y}$  of  $\mathbb{Z}^3$  are 2-adjacent if  $\|\mathbf{x} - \mathbf{y}\|_1 = 1$ .*

*A subset  $A \subseteq \mathbb{Z}^3$  is 2-connected if for every  $\mathbf{x}, \mathbf{y} \in A$ , there exist  $\mathbf{x}_1, \dots, \mathbf{x}_n \in A$  such that  $\mathbf{x}_i$  and  $\mathbf{x}_{i+1}$  are 2-adjacent for all  $i \in \{1, \dots, n-1\}$ , with  $\mathbf{x}_1 = \mathbf{x}$  and  $\mathbf{x}_n = \mathbf{y}$ .*

**Definition 2.5 (Connecting thickness [9]).** *Given  $\mathbf{v} \in \mathbb{R}_+^3$  and  $\mu \in \mathbb{R}$ , the connecting thickness  $\Omega(\mathbf{v}, \mu)$  is defined by:*

$$\Omega(\mathbf{v}, \mu) = \inf \{ \omega \in \mathbb{R}_+ : \mathfrak{P}(\mathbf{v}, \mu, \omega) \text{ is 2-connected} \}.$$

In order to compute  $\Omega(\mathbf{v}, \mu)$  we may assume without loss of generality that  $0 \leq \mathbf{v}_1 \leq \mathbf{v}_2 \leq \mathbf{v}_3$ . We thus restrict ourselves in the sequel to the set of parameters  $\mathcal{O}_3^+ = \{ \mathbf{v} \in \mathbb{R}^3 : 0 \leq \mathbf{v}_1 \leq \mathbf{v}_2 \leq \mathbf{v}_3 \}$ .

A first approximation of  $\Omega(\mathbf{v}, \mu)$  is provided by  $\|\mathbf{v}\|_\infty \leq \Omega(\mathbf{v}, \mu) \leq \|\mathbf{v}\|_1$  (see Corollary 10 of [2]). Given  $\mathbf{v} \in \mathcal{O}_3^+$  and  $\varepsilon > 0$ , it follows from the definition of  $\Omega(\mathbf{v}, \mu)$  that  $\mathfrak{P}(\mathbf{v}, \mu, \Omega(\mathbf{v}) - \varepsilon)$  is not 2-connected and that  $\mathfrak{P}(\mathbf{v}, \mu, \Omega(\mathbf{v}) + \varepsilon)$  is 2-connected.

It is shown in [10] how to compute  $\Omega(\mathbf{v}, \mu)$  from the expansion of the vector  $\mathbf{v}$  according to the *ordered fully subtractive algorithm*  $\mathbf{F} : \mathcal{O}_3^+ \rightarrow \mathcal{O}_3^+$  defined by:

$$\mathbf{F}(\mathbf{v}) = \begin{cases} (\mathbf{v}_1, \mathbf{v}_2 - \mathbf{v}_1, \mathbf{v}_3 - \mathbf{v}_1) & \text{if } \mathbf{v}_1 \leq \mathbf{v}_2 - \mathbf{v}_1 \leq \mathbf{v}_3 - \mathbf{v}_1 \\ (\mathbf{v}_2 - \mathbf{v}_1, \mathbf{v}_1, \mathbf{v}_3 - \mathbf{v}_1) & \text{if } \mathbf{v}_2 - \mathbf{v}_1 \leq \mathbf{v}_1 \leq \mathbf{v}_3 - \mathbf{v}_1 \\ (\mathbf{v}_2 - \mathbf{v}_1, \mathbf{v}_3 - \mathbf{v}_1, \mathbf{v}_1) & \text{if } \mathbf{v}_2 - \mathbf{v}_1 \leq \mathbf{v}_3 - \mathbf{v}_1 \leq \mathbf{v}_1. \end{cases} \quad (1)$$

**Theorem 2.6 ([10]).** *Let  $\mathbf{v} \in \mathcal{O}_3^+$  and  $\mu \in \mathbb{R}$ . The arithmetical discrete plane  $\mathfrak{P}(\mathbf{v}, \mu, \omega)$  is 2-connected if and only if so is  $\mathfrak{P}(\mathbf{F}(\mathbf{v}), \mu, \omega - \mathbf{v}_1)$ . Consequently,  $\Omega(\mathbf{v}, \mu) = \Omega(\mathbf{F}(\mathbf{v}), \mu) + \mathbf{v}_1$ .*

Let us fix  $\mathbf{v} \in \mathcal{O}_3^+$ , and consider the expansion of  $\mathbf{v}$  according to the ordered fully subtractive algorithm  $\mathbf{F}$  (Eq. 1). We recover a possibly infinite sequence of matrices  $(\mathbf{M}_n)_{n \in \mathbb{N}}$  with values in  $\{\mathbf{M}_1^{\text{FS}}, \mathbf{M}_2^{\text{FS}}, \mathbf{M}_3^{\text{FS}}\}$  where

$$\mathbf{M}_1^{\text{FS}} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \quad \mathbf{M}_2^{\text{FS}} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \quad \mathbf{M}_3^{\text{FS}} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix},$$

and a sequence of nonzero vectors  $(\mathbf{v}^{(n)})_{n \in \mathbb{N}}$  with nonnegative entries such that, for all  $n \in \mathbb{N}^*$ ,  $\mathbf{v} = \mathbf{M}_1 \dots \mathbf{M}_n \cdot \mathbf{v}^{(n)}$ , with  $\mathbf{v}^{(n)} = \mathbf{F}^n(\mathbf{v})$ . We set  $\mathbf{v}^{(0)} = \mathbf{v}$ .

The set of parameters  $\mathbf{v}$  for which  $\mathbf{v}_1^{(n)} + \mathbf{v}_2^{(n)} > \mathbf{v}_3^{(n)}$  for all  $n$  has been shown in [11] to play here a particular role: indeed  $\lim_{n \rightarrow \infty} \mathbf{v}^{(n)} = \mathbf{0}$  if and only if  $\mathbf{v}_1^{(n)} + \mathbf{v}_2^{(n)} \geq \mathbf{v}_3^{(n)}$  for all  $n$ . We thus introduce the following notation:

$$\mathcal{F}_3 = \left\{ \mathbf{v} \in \mathcal{O}_3^+ : \mathbf{v}_1^{(n)} + \mathbf{v}_2^{(n)} > \mathbf{v}_3^{(n)}, \text{ for all } n \in \mathbb{N} \right\}.$$

*Remark 2.7 ([14]).* If  $\mathbf{v} \in \mathcal{F}_3$  then  $\dim_{\mathbb{Q}}\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\} = 3$ .

Let us illustrate the interest of working with this set of parameters.

**Proposition 2.8.** *If  $\mathbf{v} \in \mathcal{F}_3$ , then  $\Omega(\mathbf{v}, \mu) = \sum_{n=0}^{\infty} \mathbf{v}_1^{(n)} = \frac{\|\mathbf{v}\|_1}{2}$ .*

*Proof.* According to Theorem 2.6, we have:  $\mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3 - 2\Omega(\mathbf{v}, \mu) = \mathbf{v}_1^{(i)} + \mathbf{v}_2^{(i)} + \mathbf{v}_3^{(i)} - 2\Omega(\mathbf{v}^{(i)}, \mu)$  for all  $i \in \{1, \dots, n\}$ . Since  $\Omega(\mathbf{v}^{(n)}, \mu) \leq \|\mathbf{v}^{(n)}\|_1$  and  $\lim_{n \rightarrow \infty} \mathbf{v}^{(n)} = \mathbf{0}$ , then  $\lim_{n \rightarrow \infty} \Omega(\mathbf{v}^{(n)}, \mu) = 0$  and the result follows.  $\square$

In particular, if  $\mathbf{v} \in \mathcal{F}_3$ , then  $\Omega(\mathbf{v}, \mu)$  does not depend on  $\mu$ . Hence, from now on, we consider only  $\mathbf{v} \in \mathcal{F}_3$  and we refer to  $\Omega(\mathbf{v}, \mu)$  as  $\Omega(\mathbf{v})$ .

### 2.3 Substitutions and Dual Substitutions

Let  $\mathcal{A} = \{1, 2, 3\}$  be a finite alphabet and  $\mathcal{A}^*$  be the set of finite words over  $\mathcal{A}$ .

**Definition 2.9 (Substitution).** *A substitution over  $\mathcal{A}$  is a morphism of the free monoid  $\mathcal{A}^*$ , i.e., a function  $\sigma : \mathcal{A}^* \rightarrow \mathcal{A}^*$  with  $\sigma(uv) = \sigma(u)\sigma(v)$  for all words  $u, v \in \mathcal{A}^*$ .*

Given a substitution  $\sigma$  over  $\mathcal{A}$ , the *incidence matrix*  $\mathbf{M}_\sigma$  of  $\sigma$  is the square matrix of size  $3 \times 3$  defined by  $\mathbf{M}_\sigma = (m_{ij})$ , where  $m_{i,j}$  is the number of occurrences of the letter  $i$  in  $\sigma(j)$ . A substitution  $\sigma$  is *unimodular* if  $\det \mathbf{M}_\sigma = \pm 1$ .

**Definition 2.10 (Dual substitution [3]).** *Let  $\sigma : \{1, 2, 3\}^* \rightarrow \{1, 2, 3\}^*$  be a unimodular substitution. The dual substitution  $\mathbf{E}_1^*(\sigma)$  is defined as*

$$\mathbf{E}_1^*(\sigma)([\mathbf{x}, i]^*) = \mathbf{M}_\sigma^{-1} \mathbf{x} + \bigcup_{(p,j,s) \in \mathcal{A}^* \times \mathcal{A} \times \mathcal{A}^* : \sigma(j)=pis} [\mathbf{M}_\sigma^{-1} \ell(s), j]^*,$$

where  $\ell : w \mapsto (|w|_1, |w|_2, |w|_3) \in \mathbb{Z}^3$  is the Parikh map counting the occurrences of each letter in a word  $w$ . We extend the above definition to any union of unit faces:  $\mathbf{E}_1^*(\sigma)(P_1 \cup P_2) = \mathbf{E}_1^*(\sigma)(P_1) \cup \mathbf{E}_1^*(\sigma)(P_2)$ .

Note that  $\mathbf{E}_1^*(\sigma \circ \sigma') = \mathbf{E}_1^*(\sigma') \circ \mathbf{E}_1^*(\sigma)$  for unimodular  $\sigma$  and  $\sigma'$  (see [3]).

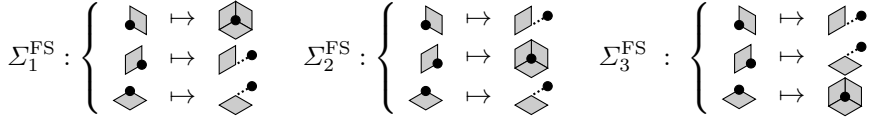
**Proposition 2.11 ([3,8]).** *We have  $\mathbf{E}_1^*(\sigma)(\Gamma_{\mathbf{v}}) = \Gamma_{\mathbf{t}_{\mathbf{M}_\sigma \mathbf{v}}}$  for every stepped plane  $\Gamma_{\mathbf{v}}$  and unimodular substitution  $\sigma$ . Furthermore, the images of two distinct faces of  $\Gamma_{\mathbf{v}}$  have no common unit face.*

We now introduce the substitutions associated with the ordered fully subtractive algorithm, which will be our main tool. Let

$$\sigma_1^{\text{FS}} = \begin{cases} 1 \mapsto 1 \\ 2 \mapsto 21 \\ 3 \mapsto 31 \end{cases} \quad \sigma_2^{\text{FS}} = \begin{cases} 1 \mapsto 2 \\ 2 \mapsto 12 \\ 3 \mapsto 32 \end{cases} \quad \sigma_3^{\text{FS}} = \begin{cases} 1 \mapsto 3 \\ 2 \mapsto 13 \\ 3 \mapsto 23 \end{cases}$$

The matrices occurring in the expansion of  $\mathbf{v}$  according to the ordered fully subtractive algorithm are the transposes of the matrices of incidence of the  $\sigma_i^{\text{FS}}$ , that is,  $\mathbf{M}_{\sigma_i^{\text{FS}}} = {}^t \mathbf{M}_i^{\text{FS}}$  for  $i \in \{1, 2, 3\}$ .

We denote by  $\Sigma_i^{\text{FS}}$  the three dual substitutions  $\mathbf{E}_1^*(\sigma_i^{\text{FS}})$  for  $i \in \{1, 2, 3\}$ . They can be represented as follows, where the black dot respectively stands for the distinguished vector of a face and its image.



### 3 Properties of the Patterns $\mathbf{T}_n$

Let  $(\mathbf{M}_n)_{n \in \mathbb{N}^*}$  be the sequence of matrices with values in  $\{\mathbf{M}_1^{\text{FS}}, \mathbf{M}_2^{\text{FS}}, \mathbf{M}_3^{\text{FS}}\}$  such that  $\mathbf{v} = \mathbf{M}_1 \cdots \mathbf{M}_n \cdot \mathbf{F}^n(\mathbf{v}) = \mathbf{M}_1 \cdots \mathbf{M}_n \cdot \mathbf{v}_n$  for all  $n$ . Let  $(\sigma_n)_{n \in \mathbb{N}^*}$  be the sequence of corresponding substitutions with values in  $\{\sigma_1^{\text{FS}}, \sigma_2^{\text{FS}}, \sigma_3^{\text{FS}}\}$ , such that  $\mathbf{M}_n = {}^t\mathbf{M}_{\sigma_n}$  for all  $n$ .

**Definition 3.1 (Generation by translations).** Let  $(\mathbf{T}_n)_{n \in \mathbb{N}}$  be the sequence of subsets of  $\mathbb{Z}^3$  defined as follows for all nonnegative integer  $n$ :

$$\mathbf{T}_0 = \{\mathbf{0}\}, \quad \mathbf{T}_1 = \{\mathbf{0}, \mathbf{e}_1\}, \quad \mathbf{T}_{n+1} = \mathbf{T}_n \cup \left( \mathbf{T}_n + {}^t(\mathbf{M}_1 \cdots \mathbf{M}_n)^{-1} \cdot \mathbf{e}_1 \right).$$

Note that the second initial condition  $\mathbf{T}_1 = \{\mathbf{0}, \mathbf{e}_1\}$  is consistent with the usual convention that an empty product of matrices is equal to the identity matrix.

**Proposition 3.2.** We have  $\cup_{n=0}^{\infty} \mathbf{T}_n \subseteq \mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$ .

*Proof (Sketch).* We prove by induction that  $\langle \mathbf{x}, \mathbf{v} \rangle < \sum_{i=0}^n \mathbf{v}_1^{(i)}$  for all  $\mathbf{x} \in \mathbf{T}_n$ , by noticing that  $\langle {}^t(\mathbf{M}_1 \cdots \mathbf{M}_n)^{-1} \cdot \mathbf{e}_1, \mathbf{v} \rangle = \langle {}^t(\mathbf{M}_1 \cdots \mathbf{M}_n)^{-1} \cdot \mathbf{e}_1, \mathbf{M}_1 \cdots \mathbf{M}_n \cdot \mathbf{v}^{(n)} \rangle = \langle \mathbf{e}_1, \mathbf{v}^{(n)} \rangle = \mathbf{v}_1^{(n)}$ . □

**Proposition 3.3.** For all  $n \in \mathbb{N}$ , the set  $\mathbf{T}_n$  is 2-connected.

*Proof (Sketch).* With the same arguments as in proof of Proposition 3.2, and by using Remark 2.7, we first get by induction that, for all  $n \in \mathbb{N}^*$ :

$$\mathbf{T}_n = \left\{ \mathbf{x} \in \mathbb{Z}^3 : \langle \mathbf{x}, \mathbf{v} \rangle = \sum_{i=0}^{n-1} \varepsilon_i \mathbf{v}_1^{(i)} \text{ with } \varepsilon_i \in \{0, 1\} \text{ for all } i \right\}.$$

Now, for all  $n \in \mathbb{N}$ , let  $\mathbf{x}_n \in \mathbf{T}_n$  such that  $\langle \mathbf{x}_n, \mathbf{v} \rangle = \sum_{i=0}^{n-1} \mathbf{v}_1^{(i)}$  (we set  $\mathbf{x}_0 = \mathbf{0}$ ). Let us prove by induction the following property: for all  $n \in \mathbb{N}^*$  there exists  $i_n \in \{1, 2, 3\}$  such that  $\mathbf{x}_n - \mathbf{e}_{i_n} \in \mathbf{T}_{n-1}$ . This property implies that  $\mathbf{x}_n$  is 2-adjacent to  $\mathbf{T}_{n-1}$ , which implies the 2-connectedness of  $\mathbf{T}_n$ .

The induction property is true for  $n = 1$  with  $\mathbf{x}_1 = \mathbf{e}_1$ . Let us now assume that the induction hypothesis hold for  $n \geq 1$ . Let  $u_1 \cdots u_n \in \{1, 2, 3\}^{\mathbb{N}^*}$  be such

that  $\mathbf{M}_{u_1}^{\text{FS}} \cdots \mathbf{M}_{u_n}^{\text{FS}} \mathbf{v}^{(n)} = \mathbf{v}$ . We have  $\langle \mathbf{x}_{n+1}, \mathbf{v} \rangle = \langle \mathbf{x}_n, \mathbf{v} \rangle + \mathbf{v}_1^{(n)}$ , and by definition of the fully subtractive algorithm (see Eq. 1):

$$\mathbf{v}_1^{(n)} = \begin{cases} \mathbf{v}_1^{(n-1)}, & \text{if } u_n = 1 \\ \mathbf{v}_2^{(n-1)} - \mathbf{v}_1^{(n-1)}, & \text{if } u_n \in \{2, 3\}. \end{cases}$$

We distinguish several cases according to the values taken by  $u_1 \cdots u_n$ .

**Case 1.** If  $u_n = 1$ , then,  $\langle \mathbf{x}_{n+1}, \mathbf{v} \rangle = \langle \mathbf{x}_n, \mathbf{v} \rangle + \mathbf{v}_1^{(n-1)}$ , and

$$\langle \mathbf{x}_{n+1} - \mathbf{e}_{i_n}, \mathbf{v} \rangle = \underbrace{\langle \mathbf{x}_n - \mathbf{e}_{i_n}, \mathbf{v} \rangle}_{\in \mathbf{T}_{n-1}} + \mathbf{v}_1^{(n-1)} = \sum_{i=1}^{n-2} \varepsilon_i \mathbf{v}_1^{(i)} + \mathbf{v}_1^{(n-1)},$$

where  $\varepsilon_i \in \{0, 1\}$  for  $1 \leq i \leq n-2$ , which implies that  $\mathbf{x}_{n+1} - \mathbf{e}_{i_n} \in \mathbf{T}_n$ , so taking  $i_{n+1} = i_n$  yields the desired result.

**Case 2.** If  $u_n \in \{2, 3\}$  and  $u_1 \cdots u_{n-1} = 1^k$ , then

$$\begin{aligned} \langle \mathbf{x}_{n+1}, \mathbf{v} \rangle &= \langle \mathbf{x}_n, \mathbf{v} \rangle + \mathbf{v}_2^{(n-1)} - \mathbf{v}_1^{(n-1)} = \langle \mathbf{x}_{n-1}, \mathbf{v} \rangle + \mathbf{v}_2^{(n-1)} \\ &= \langle \mathbf{x}_{n-2}, \mathbf{v} \rangle + \mathbf{v}_2^{(n-2)} = \cdots = \langle \mathbf{x}_{n-1-k}, \mathbf{v} \rangle + \mathbf{v}_2^{(n-1-k)} = \mathbf{v}_2^{(0)}, \end{aligned}$$

which implies that  $\mathbf{x}_{n+1} - \mathbf{e}_2 \in \mathbf{T}_n$ .

**Case 3.** If  $u_n \in \{2, 3\}$  and  $u_1 \cdots u_{n-1} = \cdots 21^k$  with  $0 \leq k \leq n-2$ , then

$$\begin{aligned} \langle \mathbf{x}_{n+1}, \mathbf{v} \rangle &= \langle \mathbf{x}_n, \mathbf{v} \rangle + \mathbf{v}_2^{(n-1)} - \mathbf{v}_1^{(n-1)} = \langle \mathbf{x}_{n-1}, \mathbf{v} \rangle + \mathbf{v}_2^{(n-1)} \\ &= \langle \mathbf{x}_{n-1-k}, \mathbf{v} \rangle + \mathbf{v}_2^{(n-1-k)} = \langle \mathbf{x}_{n-1-k}, \mathbf{v} \rangle + \mathbf{v}_1^{(n-2-k)}, \end{aligned}$$

so  $\mathbf{x}_{n+1} - \mathbf{e}_{i_{n-1-k}} \in \mathbf{T}_{n-1-k} \subseteq \mathbf{T}_n$ .

**Case 4.** If  $u_n \in \{2, 3\}$  and  $u_1 \cdots u_{n-1} = w31^k$  with  $w \in \{1, 2\}^\ell$  and  $k \geq 0$ , then

$$\begin{aligned} \langle \mathbf{x}_{n+1}, \mathbf{v} \rangle &= \langle \mathbf{x}_{n-1-k}, \mathbf{v} \rangle + \mathbf{v}_2^{(n-1-k)} = \langle \mathbf{x}_{n-2-k}, \mathbf{v} \rangle + \mathbf{v}_3^{(n-2-k)} \\ &= \langle \mathbf{x}_{n-2-k-\ell}, \mathbf{v} \rangle + \mathbf{v}_3^{(n-2-k-\ell)} = \mathbf{v}_3^{(0)}, \end{aligned}$$

so  $\mathbf{x}_{n+1} - \mathbf{e}_3 \in \mathbf{T}_n$ .

**Case 5.** If  $u_n \in \{2, 3\}$  and  $u_1 \cdots u_{n-1} = \cdots 3w31^k$  with  $w \in \{1, 2\}^\ell$ ,  $k \geq 0$ , then

$$\begin{aligned} \langle \mathbf{x}_{n+1}, \mathbf{v} \rangle &= \langle \mathbf{x}_{n-2-k-\ell}, \mathbf{v} \rangle + \mathbf{v}_3^{(n-2-k-\ell)} \\ &= \langle \mathbf{x}_{n-2-k-\ell}, \mathbf{v} \rangle + \mathbf{v}_1^{(n-3-k-\ell)} \end{aligned}$$

so  $\mathbf{x}_{n+1} - \mathbf{e}_{i_{n-2-k-\ell}} \in \mathbf{T}_{n-2-k-\ell} \subseteq \mathbf{T}_n$ . □

**Definition 3.4 (Generation by dual substitutions).** Let  $\mathcal{U} = [0, 1]^* \cup [0, 2]^* \cup [0, 3]^* = \blacklozenge$  and let  $(\sigma_n)_{n \in \mathbb{N}^*}$  be the sequence of fully subtractive substitutions generated by  $\mathbf{v}$ . For  $n \in \mathbb{N}$ , let:

- $P_n = \mathbf{E}_1^*(\sigma_n \circ \cdots \circ \sigma_1)(\mathcal{U})$  with  $P_0 = \mathcal{U}$ ;



–  $\mathbf{P}_n$  be the set of distinguished vertices of the set  $P_n$  of unit faces.

**Proposition 3.5.** *For every  $n \in \mathbb{N}$ , we have  $\mathbf{P}_n \subseteq \mathbf{T}_n$ .*

*Proof.* We first remark that  $\mathbf{E}_1^*(\sigma_i^{\text{FS}})(\mathcal{U}) = \mathcal{U} \cup [\mathbf{e}_1, 2]^* \cup [\mathbf{e}_1, 3]^* = \text{img}$  for all  $i \in \{1, 2, 3\}$ . For  $n \in \mathbb{N}$ , we have

$$\begin{aligned} P_{n+1} &= \mathbf{E}_1^*(\sigma_{n+1} \circ \dots \circ \sigma_1)(\mathcal{U}) \\ &= \mathbf{E}_1^*(\sigma_n \circ \dots \circ \sigma_1) \circ \mathbf{E}_1^*(\sigma_{n+1})(\mathcal{U}) \\ &= P_n \cup \mathbf{E}_1^*(\sigma_n \circ \dots \circ \sigma_1)([\mathbf{e}_1, 2]^* \cup [\mathbf{e}_1, 3]^*), \end{aligned}$$

which implies  $P_n \subseteq P_{n+1}$ . Since  $[\mathbf{e}_1, 2]^* \cup [\mathbf{e}_1, 3]^* \subseteq \mathbf{e}_1 + \mathcal{U}$ , we have  $P_{n+1} \subseteq P_n \cup \mathbf{E}_1^*(\sigma_n \circ \dots \circ \sigma_1)(\mathbf{e}_1 + \mathcal{U})$ . By Definition 2.10, we then have

$$\begin{aligned} \mathbf{E}_1^*(\sigma_n \circ \dots \circ \sigma_1)(\mathbf{e}_1 + \mathcal{U}) &= \mathbf{M}_{\sigma_n \circ \dots \circ \sigma_1}^{-1} \cdot \mathbf{e}_1 + \mathbf{E}_1^*(\sigma_n \circ \dots \circ \sigma_1)(\mathcal{U}) \\ &= ({}^t\mathbf{M}_n \cdots {}^t\mathbf{M}_1)^{-1} \cdot \mathbf{e}_1 + P_n \\ &= {}^t(\mathbf{M}_1 \cdots \mathbf{M}_n)^{-1} \cdot \mathbf{e}_1 + P_n, \end{aligned}$$

which proves  $P_n \subseteq P_{n+1} \subseteq P_n \cup (P_n + {}^t(\mathbf{M}_1 \cdots \mathbf{M}_n)^{-1} \cdot \mathbf{e}_1)$ . The result now follows by induction. □

## 4 Generation of Naive Planes with Dual Substitutions

The aim of this section is to prove that the patterns  $\mathbf{P}_n$  cover the naive plane, by showing that iterations of dual substitutions yield concentric annuli (see Definition 4.5) with increasing radius. The main result of this section is the following.

**Proposition 4.1.** *If  $\mathbf{v} \in \mathcal{F}_3$ , then  $\bigcup_{n=0}^\infty \mathbf{P}_n = \mathfrak{P}(\mathbf{v}, 0, \|\mathbf{v}\|_\infty)$ .*

The proof will be given at the end of Section 4.3. The remaining of this section is devoted to the development of specific tools used in this proof. Such tools have also been used in [5] to study other multidimensional continued fraction algorithms.

### 4.1 Covering Properties and Annuli

A *pattern* is a union of unit faces. In the rest of this section we will consider some sets of connected patterns ( $\mathcal{L}$ ,  $\mathcal{L}_{\text{edge}}$  and  $\mathcal{L}_{\text{FS}}$ ) that will be needed in order to define (strong) coverings. The patterns contained in these sets are considered up to translation only, as it is all that matters for the definitions below (see Figure 1).

**Definition 4.2 ( $\mathcal{L}$ -cover).** *Let  $\mathcal{L}$  be a set of patterns. A pattern  $P$  is  $\mathcal{L}$ -covered if for all faces  $e, f \in P$ , there exist  $Q_1, \dots, Q_n \in \mathcal{L}$  such that:*

1.  $e \in Q_1$  and  $f \in Q_n$ ;
2.  $Q_k \cap Q_{k+1}$  contains at least one face, for all  $k \in \{1, \dots, n-1\}$ ;
3.  $Q_k \subseteq P$  for all  $k \in \{1, \dots, n\}$ .

**Lemma 4.3.** *Let  $P$  be an  $\mathcal{L}$ -covered pattern,  $\Sigma$  a dual substitution and  $\mathcal{L}$  a set of patterns such that  $\Sigma(Q)$  is  $\mathcal{L}$ -covered for all  $Q \in \mathcal{L}$ . Then  $\Sigma(P)$  is  $\mathcal{L}$ -covered.*

We will need *strong* coverings to ensure that the image of an annulus is an annulus. We denote by  $\mathcal{L}_{\text{edge}}$  the set of all the twelve edge-connected two-face patterns (up to translation).

**Definition 4.4 (Strong  $\mathcal{L}$ -cover).** *Let  $\mathcal{L}$  be a set of edge-connected patterns. A pattern  $P$  is strongly  $\mathcal{L}$ -covered if*

1.  $P$  is  $\mathcal{L}$ -covered;
2. for every pattern  $X \in \mathcal{L}_{\text{edge}}$  such that  $X \subseteq P$ , there exists a pattern  $Y \in \mathcal{L}$  such that  $X \subseteq Y \subseteq P$ .

The intuitive idea behind the notion of strong  $\mathcal{L}$ -covering is that every occurrence of a pattern of  $\mathcal{L}_{\text{edge}}$  in  $P$  is required to be “completed within  $P$ ” by a pattern of  $\mathcal{L}$ .

**Definition 4.5 (Annulus).** *Let  $\mathcal{L}$  be a set of edge-connected patterns and  $\Gamma$  be a stepped plane. An  $\mathcal{L}$ -annulus of a pattern  $P \subseteq \Gamma$  is a pattern  $A \subseteq \Gamma$  such that:*

1.  $P$ ,  $A \cup P$  and  $\Gamma \setminus (A \cup P)$  are  $\mathcal{L}$ -covered;
2.  $A$  is strongly  $\mathcal{L}$ -covered;
3.  $A$  and  $P$  have no face in common;
4.  $P \cap \overline{\Gamma \setminus (P \cup A)} = \emptyset$ .

The notation  $\overline{\Gamma \setminus (P \cup A)}$  stands for the topological closure of  $\Gamma \setminus (P \cup A)$ .

Conditions 1 and 2 are combinatorial properties that we will use in the proof of Lemma 4.11 in order to prove that the image of an  $\mathcal{L}_{\text{FS}}$ -annulus by a  $\Sigma_i^{\text{FS}}$  is an  $\mathcal{L}_{\text{FS}}$ -annulus. Conditions 3 and 4 are properties of topological nature that we want annuli to satisfy.

## 4.2 Covering Properties for $\Sigma_1, \Sigma_2, \Sigma_3$

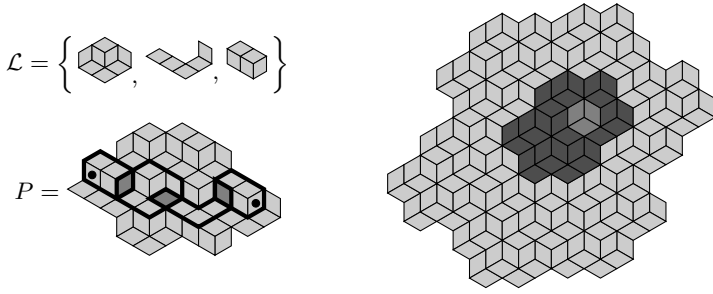
Let  $\mathcal{L}_{\text{FS}}$  be the set of patterns containing  $\begin{smallmatrix} \blacktriangleleft \\ \blacktriangleright \end{smallmatrix}, \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix}, \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleright \end{smallmatrix}, \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix}, \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix}, \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix} \in \mathcal{L}_{\text{edge}}$  and

$$\begin{aligned} \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix} &= [0, 2]^* \cup [(1, 0, 0), 2]^* \cup [0, 3]^*, \\ \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix} &= [0, 3]^* \cup [(1, 0, 0), 3]^* \cup [0, 2]^*, \\ \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix} &= [0, 3]^* \cup [(0, 1, 0), 3]^* \cup [0, 1]^*. \end{aligned}$$

**Lemma 4.6 ( $\mathcal{L}_{\text{FS}}$ -covering).** *Let  $P$  be an  $\mathcal{L}_{\text{FS}}$ -covered pattern. Then the pattern  $\Sigma_i(P)$  is  $\mathcal{L}_{\text{FS}}$ -covered for every  $i \in \{1, 2, 3\}$ .*

**Lemma 4.7.** *Let  $\Gamma$  be a stepped plane that does not contain any translate of one of the patterns  $\begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix}, \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix}, \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix} \in \mathcal{L}_{\text{edge}}$  and  $\begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix} = [0, 3]^* \cup [(1, 1, 0), 3]^*$ . Then no translate of any of these four patterns appears in  $\Sigma_i(\Gamma)$ .*

**Lemma 4.8 (Strong  $\mathcal{L}_{\text{FS}}$ -covering).** *Let  $P$  be a strongly  $\mathcal{L}_{\text{FS}}$ -covered pattern which is contained in a stepped plane that avoids  $\begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix}, \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix}, \begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix}$  and  $\begin{smallmatrix} \blacktriangleleft \\ \blacktriangleleft \end{smallmatrix}$ . Then  $\Sigma_i(P)$  is strongly  $\mathcal{L}_{\text{FS}}$ -covered for every  $i \in \{1, 2, 3\}$ .*



**Fig. 1.** On the left, the pattern  $P$  is  $\mathcal{L}$ -covered. Two faces of  $P$  are connected via a sequence of patterns from  $\mathcal{L}$ . On the right, examples of  $\mathcal{L}_{\text{FS}}$ -annulus. Patterns  $P_0 \subsetneq P_4 \subsetneq P_7$  are defined by  $P_0 = \mathcal{U}$  and  $P_{i+1} = \Sigma_3^{\text{FS}}(P_i)$ . The lighter pattern  $P_7 \setminus P_4$  is a  $\mathcal{L}_{\text{FS}}$ -annulus of  $P_4$  and the darker pattern  $P_4 \setminus P_0$  is an  $\mathcal{L}_{\text{FS}}$ -annulus of  $P_0$ .

### 4.3 Annuli and Dual Substitutions

The proof of the following proposition (by induction) relies on Lemma 4.10 (base case) and on Lemma 4.11 (induction step). We recall that  $\mathcal{U} = [0, 1]^* \cup [0, 2]^* \cup [0, 3]^* = \text{[cube]}$

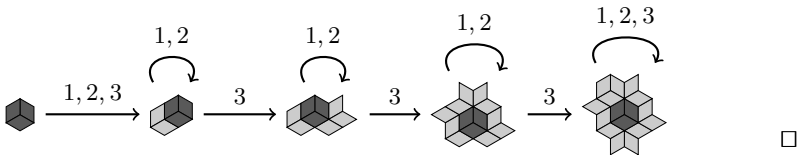
**Proposition 4.9.** *Let  $(\Sigma_i)_{i \in \mathbb{N}}$  be a sequence with values in  $\{\Sigma_1^{\text{FS}}, \Sigma_2^{\text{FS}}, \Sigma_3^{\text{FS}}\}$  such that  $\Sigma_3^{\text{FS}}$  occurs infinitely often, and let  $k \in \mathbb{N}$  such that  $(\Sigma_1, \dots, \Sigma_k)$  contains  $\Sigma_3^{\text{FS}}$  at least four times.*

*Then for every  $\ell \geq 1$ ,  $\Sigma_1 \cdots \Sigma_{k+\ell}(\mathcal{U}) \setminus \Sigma_1 \cdots \Sigma_\ell(\mathcal{U})$  is an  $\mathcal{L}_{\text{FS}}$ -annulus of  $\Sigma_1 \cdots \Sigma_\ell(\mathcal{U})$  in the stepped plane  $\Sigma_1 \cdots \Sigma_{k+\ell}(\mathbf{\Gamma}_{(1,1,1)})$ .*

*Proof.* We prove the result by induction on  $\ell$ . The case  $\ell = 0$  (i.e.,  $\Sigma_1 \cdots \Sigma_k(\mathcal{U}) \setminus \mathcal{U}$  is an annulus of  $\mathcal{U}$ ) is settled by Lemma 4.10. Now, assume that the induction property holds for some  $\ell \in \mathbb{N}$ . The pattern  $\Sigma_1 \cdots \Sigma_{k+\ell}(\mathcal{U})$  is contained in the stepped plane  $\Sigma_{k+\ell}(\mathbf{\Gamma}_{(1,1,1)})$ , so it does not contain any of the patterns forbidden by Lemma 4.7. We can then apply Lemma 4.11 to deduce that  $\Sigma_1 \cdots \Sigma_{k+\ell+1}(\mathcal{U}) \setminus \Sigma_1 \cdots \Sigma_{k+\ell}(\mathcal{U})$  is an  $\mathcal{L}_{\text{FS}}$ -annulus of  $\Sigma_1 \cdots \Sigma_{k+\ell}(\mathcal{U})$ .  $\square$

**Lemma 4.10.** *Let  $\Sigma$  be a product of  $\Sigma_1$ ,  $\Sigma_2$  and  $\Sigma_3$  such that  $\Sigma_3$  appears at least four times. Then  $\Sigma(\mathcal{U}) \setminus \mathcal{U}$  is an  $\mathcal{L}_{\text{FS}}$ -annulus of  $\mathcal{U}$  in  $\Sigma(\mathbf{\Gamma}_{(1,1,1)})$ .*

*Proof.* Below, “ $P \xrightarrow{i} Q$ ” means that  $Q \subseteq \Sigma_i(P)$  so the result follows.



**Lemma 4.11.** *Let  $\mathbf{\Gamma}$  be a stepped plane that avoids  $\text{[forbidden patterns]}$ . Let  $A \subseteq \mathbf{\Gamma}$  be an  $\mathcal{L}_{\text{FS}}$ -annulus of a pattern  $P \subseteq \mathbf{\Gamma}$ , and let  $\Sigma = \Sigma_i$  for some  $i \in \{1, 2, 3\}$ . Then  $\Sigma(A)$  is an  $\mathcal{L}_{\text{FS}}$ -annulus of  $\Sigma(P)$  in the stepped plane  $\Sigma(\mathbf{\Gamma})$ .*



*Proof.* We must prove the following:

1.  $\Sigma(P)$ ,  $\Sigma(A) \cup \Sigma(P)$  and  $\Gamma \setminus (\Sigma(A) \cup \Sigma(P))$  are  $\mathcal{L}_{\text{FS}}$ -covered;
2.  $\Sigma(A)$  is strongly  $\mathcal{L}_{\text{FS}}$ -covered;
3.  $\Sigma(A)$  and  $\Sigma(P)$  have no face in common;
4.  $\Sigma(P) \cap \overline{\Sigma(\Gamma) \setminus (\Sigma(P) \cup \Sigma(A))} = \emptyset$ .

Conditions 1 and 3 hold thanks to Lemma 4.3 and Proposition 2.11 respectively, and 2 holds thanks to Lemma 4.8. It remains to prove that 4 holds.

Suppose that 4 does not hold. This implies that there exist faces  $f \in P$ ,  $g \in \Gamma \setminus (A \cup P)$ ,  $f' \in \Sigma(f)$  and  $g' \in \Sigma(g)$  such that  $f'$  and  $g'$  have a nonempty intersection. Also,  $f \cup g$  must be disconnected because  $P$  and  $\overline{\Gamma \setminus (P \cup A)}$  have empty intersection by hypothesis.

The strategy of the proof is as follows: we check all the possible patterns  $f \cup g$  and  $f' \cup g'$  as above, and for each case we derive a contradiction. This can be done by inspection of a finite number of cases. Indeed, there are 36 possibilities for  $f' \cup g'$  up to translation (the number of connected two-face patterns that share a vertex or an edge), and each of these patterns has a finite number of two-face preimages.

The first patterns  $f' \cup g'$  which have disconnected preimages are  $f' \cup g' = [0, 3]^* \cup [(1, 1, 0), 3]^*$  or  $[0, 2]^* \cup [(1, -1, 1), 1]^*$  or  $[0, 2]^* \cup [(1, 0, 1), 2]^*$ . These cases can be ignored thanks to Lemma 4.7: the first case () is forbidden by assumption. In the second case, Definition 2.2 implies that if a stepped plane contains  $f' \cup g'$ , then it contains the face  $[(0, 0, 1), 2]^*$  shown in dark gray . This contains a pattern ruled out by Lemma 4.7, which settles this case. The third case can be treated in the same way.




Another possibility is  $f' \cup g' = [0, 2]^* \cup [(1, -1, 1), 3]^*$ , which admits six disconnected preimages (two for each  $\Sigma_i$ ). They are shown below (in light gray), together with their only possible completion within a stepped plane (in dark gray), which can be deduced from Definition 2.2:



The patterns that appear in dark gray are forbidden by Lemma 4.7, so this case is settled.

The last two possibilities are  $f' \cup g' = [0, 3]^* \cup [(1, 1, -1), 1]^*$  or  $f' \cup g' = [0, 3]^* \cup [(1, 1, -1), 2]^*$ . Below (in light gray) are all the possible preimages  $f \cup g$  (which are the same for the two possibilities), and in dark gray is shown their only possible completion  $X$  within a stepped plane:



Now, we have  $X \subseteq A$  because Condition 4 for  $A$  and  $P$  would fail otherwise ( $f$  and  $g$  cannot touch). However, this contradicts the fact that strongly  $\mathcal{L}_{\text{FS}}$ -connected. Indeed,  $X \in \mathcal{L}_{\text{edge}}$  but there cannot exist a pattern  $Y \in \mathcal{L}_{\text{FS}}$  such that  $X \subseteq Y \subseteq A$  because then we must have  $Y =$  ,  or , so  $Y$  must overlap with  $f$  or  $g$ , which is impossible because  $f$  and  $g$  are not in  $A$ .  $\square$

*Proof (Proof of Proposition 4.1).* Let  $\mathbf{v} \in \mathcal{F}_3$ . To prove the proposition, it is enough to prove that  $\cup_{n=0}^\infty P_n = \Gamma_{\mathbf{v}}$ , thanks to Remark 2.3. Let  $P \in \Gamma_{\mathbf{v}}$  be a finite pattern. The *combinatorial radius* of  $P$  is defined to be the length of the smallest path of edge-connected unit faces from the origin to  $\Gamma_{\mathbf{v}} \setminus P$ .

Now, since  $\mathbf{v} \in \mathcal{F}_3$ ,  $\Sigma_3^{\text{FS}}$  occurs infinitely many often in the sequence  $(\Sigma_i)_{i \in \mathbb{N}}$  of the dual substitutions associated with the expansion of  $\mathbf{v}$ . Hence, we can apply Proposition 4.9 to prove that there exists  $k \in \mathbb{N}$  such that for all  $\ell \geq 0$ , the pattern  $A_\ell := \Sigma_1 \cdots \Sigma_{k+\ell}(\mathcal{U}) \setminus \Sigma_1 \cdots \Sigma_\ell(\mathcal{U})$  is an  $\mathcal{L}_{\text{FS}}$ -annulus of  $\Sigma_1 \cdots \Sigma_\ell(\mathcal{U})$ .

By Condition 1 of Definition 4.5, the pattern  $A_\ell \cup \Sigma_1 \cdots \Sigma_\ell(\mathcal{U})$  is simply connected for all  $\ell \geq 0$ , so its combinatorial radius increases at least by 1 when  $\ell$  is incremented, thanks to Conditions 3 4. This proves the required property.  $\square$

## 5 Main Results

In general, we do not know if the set  $\{\omega \in \mathbb{R} \mid \mathfrak{P}(\mathbf{v}, \omega, \Omega(\mathbf{v})) \text{ is 2-connected}\}$  is closed. In fact,  $\{\omega \in \mathbb{R} \mid \mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))\}$  may be 2-connected or not.

**Theorem 5.1.** *Let  $\mathbf{v} \in \mathcal{O}_3^+$ . The arithmetical discrete plane  $\mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$  is 2-connected if and only if  $\mathbf{v} \in \mathcal{F}_3$ .*

*Proof.* Let  $\mathbf{v} \in \mathcal{F}_3$  and  $\mathbf{x} \in \mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$ , by Proposition 2.8 we have  $\Omega(\mathbf{v}) = \|\mathbf{v}\|_1/2$ . If  $\|\mathbf{v}\|_\infty \leq \langle \mathbf{x}, \mathbf{v} \rangle < \|\mathbf{v}\|_1/2$ , then  $\|\mathbf{v}\|_\infty - \mathbf{v}_1 \leq \langle \mathbf{x} - \mathbf{e}_1, \mathbf{v} \rangle < \|\mathbf{v}\|_1/2 - \mathbf{v}_1 < \|\mathbf{v}\|_\infty$ , so  $\mathbf{x} - \mathbf{e}_1 \in \mathfrak{P}(\mathbf{v}, 0, \|\mathbf{v}\|_\infty)$ . In other words, an element  $\mathbf{x}$  of  $\mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$  either belongs to  $\mathfrak{P}(\mathbf{v}, 0, \|\mathbf{v}\|_\infty)$  or is 2-adjacent to an element of  $\mathfrak{P}(\mathbf{v}, 0, \|\mathbf{v}\|_\infty)$ .

Now, given  $\mathbf{y} \in \mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$ , both  $\mathbf{x}$  and  $\mathbf{y}$  belong or are adjacent to  $\mathfrak{P}(\mathbf{v}, 0, \|\mathbf{v}\|_\infty)$ , so they are 2-connected in  $\mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$  because:

- $\mathfrak{P}(\mathbf{v}, 0, \|\mathbf{v}\|_\infty) \subseteq \cup_{n=0}^\infty \mathbf{T}_n$ , thanks Propositions 3.5 and 4.1,
- $\cup_{n=0}^\infty \mathbf{T}_n$  is 2-connected: it is a increasing union of sets  $\mathbf{T}_n$  which are 2-connected thanks to Proposition 3.3,
- $\cup_{n=0}^\infty \mathbf{T}_n \subseteq \mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$ , thanks to Proposition 3.2.

We now prove the converse implication, and we assume that  $\mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$  is 2-connected. Assume  $\dim_{\mathbb{Q}}\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\} = 1$ ,  $\mathbf{v} \in \mathbb{Z}^3$  with  $\gcd\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\} = 1$ . Let  $n \in \mathbb{N}$  such that  $\mathbf{v}_1^{(n)} = 0$ . The plane  $\mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$  is 2-connected if and only if so is  $\mathfrak{P}(\mathbf{v}^{(n)}, 0, \Omega(\mathbf{v}^{(n)}))$ . But  $\Omega(\mathbf{v}^{(n)}) = \mathbf{v}_2^{(n)} + \mathbf{v}_3^{(n)} - 1$  so  $\mathfrak{P}(\mathbf{v}^{(n)}, 0, \Omega(\mathbf{v}^{(n)}))$  is the translation along  $\mathbf{e}_1$  of an arithmetical discrete line strictly thinner than a standard one and cannot be 2-connected. Hence  $\dim_{\mathbb{Q}}\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\} > 1$ . If  $\mathbf{v} \notin \mathcal{F}_3$ , there exists  $n \in \mathbb{N}$  such that  $\mathbf{v}_1^{(n)} + \mathbf{v}_2^{(n)} \leq \mathbf{v}_3^{(n)}$ . The plane  $\mathfrak{P}(\mathbf{v}, 0, \Omega(\mathbf{v}))$  is 2-connected if and only if so is  $\mathfrak{P}(\mathbf{v}^{(n)}, 0, \Omega(\mathbf{v}^{(n)}))$ . But  $\Omega(\mathbf{v}^{(n)}) = \|\mathbf{v}^{(n)}\|_\infty$ , so  $\mathfrak{P}(\mathbf{v}^{(n)}, 0, \|\mathbf{v}^{(n)}\|_\infty)$  cannot be 2-connected since  $\mathbf{x}$  and  $\mathbf{x} + \mathbf{e}_3$  cannot be both in  $\mathfrak{P}(\mathbf{v}^{(n)}, 0, \|\mathbf{v}^{(n)}\|_\infty)$ .  $\square$

The theorem below is a direct consequence of Propositions 3.5 and 4.1, but it is worth mentioning.

**Theorem 5.2.** *If  $\mathbf{v} \in \mathcal{F}_3$ , then  $\mathfrak{P}(\mathbf{v}, 0, \|\mathbf{v}\|_\infty) \subseteq \cup_{n=0}^\infty \mathbf{T}_n$ , i.e., the naive plane of normal vector  $\mathbf{v}$  is included in  $\cup_{n=0}^\infty \mathbf{T}_n$ .*

## 6 Conclusion

We have provided a full understanding of the 2-connectedness of discrete planes with critical thickness, using a combination of tools issued from numeration systems and combinatorics on words. Theorem 5.1 highlights the limit behavior of discrete plane with critical thickness which is reminiscent of similar phenomena occurring in percolation theory. We plan to investigate further the properties of critical planes (as in [4]), their tree structure, and we plan to explore more deeply the connections with Rauzy fractals and numeration systems.

**Acknowledgments.** The authors warmly thank Éric Domenjoud and Laurent Vuillon for many helpful discussions.

## References

1. Andres, E.: Discrete linear objects in dimension  $n$ : the standard model. *Graphical Models* 65(1-3), 92–111 (2003)
2. Andres, E., Acharya, R., Sibata, C.: Discrete analytical hyperplanes. *CVGIP: Graphical Model and Image Processing* 59(5), 302–309 (1997)
3. Arnoux, P., Ito, S.: Pisot substitutions and Rauzy fractals. *Bull. Belg. Math. Soc. Simon Stevin* 8(2), 181–207 (2001)
4. Berthé, V., Domenjoud, E., Jamet, D., Provençal, X., Toutant, J.L.: Oral Communication in Numeration and Substitution 2012, June 4-8, Kyoto (2012)
5. Berthé, V., Jolivet, T., Siegel, A.: Substitutive Arnoux-rauzy sequences have pure discrete spectrum. *Unif. Distrib. Theory* 7(1), 173–197 (2012)
6. Brimkov, V.E., Barneva, R.P.: Connectivity of discrete planes. *Theor. Comput. Sci.* 319(1-3), 203–227 (2004)
7. 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)
8. Fernique, T.: Bidimensional Sturmian Sequences and Substitutions. In: De Felice, C., Restivo, A. (eds.) *DLT 2005*. LNCS, vol. 3572, pp. 236–247. Springer, Heidelberg (2005)
9. Jamet, D., Toutant, J.L.: On the Connectedness of Rational Arithmetic Discrete Hyperplanes. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) *DGCI 2006*. LNCS, vol. 4245, pp. 223–234. Springer, Heidelberg (2006)
10. Jamet, D., Toutant, J.L.: Minimal arithmetic thickness connecting discrete planes. *Discrete Applied Mathematics* 157(3), 500–509 (2009)
11. Meester, R.W.J.: An algorithm for calculating critical probabilities and percolation functions in percolation models defined by rotations. *Ergodic Theory Dynam. Systems* 9(3), 495–509 (1989)
12. Meester, R.W.J., Nowicki, T.: Infinite clusters and critical values in two-dimensional circle percolation. *Israel J. Math.* 68(1), 63–81 (1989)
13. Reveillès, J.P.: *Géométrie discrète, calcul en nombres entiers et algorithmique*. Thèse d'état, Université Louis Pasteur, Strasbourg (1991)
14. Schweiger, F.: *Multidimensional continued fractions*. Oxford Science Publications. Oxford University Press, Oxford (2000)

# A 3D Curvilinear Skeletonization Algorithm with Application to Path Tracing

John Chaussard<sup>1</sup>, Laurent Noël<sup>2</sup>, Venceslas Biri<sup>2</sup>, and Michel Couprie<sup>2,\*</sup>

<sup>1</sup> Université Paris 13, Sorbonne Paris Cité, LAGA, CNRS(UMR 7539),  
F-93430, Villetaneuse, France

`chaussard@math.univ-paris13.fr`

<sup>2</sup> Université Paris Est, LIGM, A3SI-ESIEE

2, boulevard Blaise Pascal, 93162 Noisy le Grand CEDEX, France  
{`laurent.noel,v.biri,michel.couprie`}@esiee.fr

**Abstract.** We present a novel 3D curvilinear skeletonization algorithm which produces filtered skeletons without needing any user input, thanks to a new parallel algorithm based on the cubical complex framework. These skeletons are used in a modified path tracing algorithm in order to produce less noisy images in less time than the classical approach.

## 1 Introduction

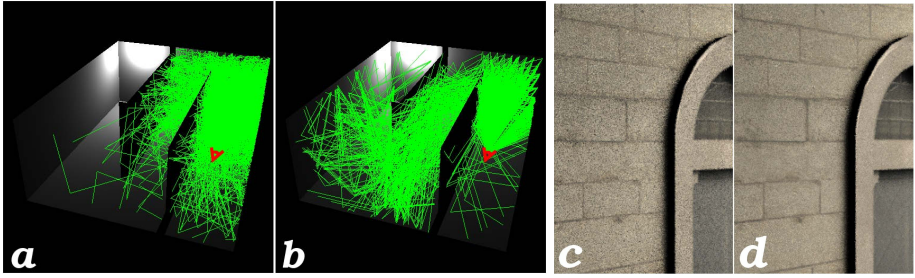
Path tracing algorithms [8,9] are able to render photorealistic images from a scene description by simulating light propagation. The photorealistic aspect is achieved by simulating diffusion of light: elements of the scene which are illuminated by a light source can cast part of the light they receive. Consequently, parts of the scene which are not directly illuminated by any light source actually receive light, thus giving a realistic effect of light diffusion to the resulting image.

A path tracing algorithm sends rays from the camera, and let them bounce around the scene until they encounter a directly illuminated zone. When a ray hits a point of any surface, it is reemitted in a random direction. Once it reaches a directly illuminated area, all luminance gathered through its journey in the scene is averaged in order to set one of the image pixels, a process repeated for each pixel of the output image. To reduce the noise resulting from this stochastic process, the algorithm actually sends many rays for each pixel of the image: the more rays, the longer the algorithm will take to produce an image, but the more realistic the result will look. A path tracing can be seen as a random walk process through the scene.

In order to avoid infinite bouncing of rays in the scene, a limit is set for the number of bounces a ray can do. If a ray does not reach any directly illuminated area after a given number of bounces, its contribution is considered null. Such lost ray is a waste of computation time, and minimizing this loss would speed up the algorithm (see Fig 1.a).

---

\* This work has been partially supported by the “ANR-2010-BLAN-0205 KIDICO” project.



**Fig. 1.** Note how rays are guided toward the illuminated area thanks to our skeleton based algorithm (image *b*), compared to classical path tracing (image *a*). Images *c* and *d* point out the noise reduction between the classic path tracing (*c*) and our method (*d*).

Our proposition is to reduce both the time taken by a path tracing algorithms and the noise in the output images by guiding rays towards the scene’s light sources (see Fig 1.b), thus reducing the number of rays “lost” during the computation. To do this, we perform some precomputation on the void space of the scene (the medium where the rays of light propagate) in order to have, later in the algorithm, a clue on the direction a ray should bounce in order to meet an illuminated area. Unfortunately, such precomputation may take time and become more expensive than the original algorithm.

To keep the precomputation fast, we perform it only on a subset of points of the voids of the scene. Since this precomputation relies on paths and visibility tests, we want this set of points to be representative of both the topology and the geometry of the original scene. For example, it should possess branches propagating in the elongated parts of the scene. A filtered curvilinear skeleton of the scene’s void meets all the required properties for our subset of points: it is a thin subset of points, with the same topology and retaining the main geometrical information of the scene. Note that in this application, the filtering of the skeleton (elimination of spurious branches) need to be both efficient and completely automatic, for one cannot ask a user to set parameters for each scene.

The purpose of this paper is twofold: we first introduce a new curvilinear skeletonization algorithm (Sec. 3) in the cubical complex framework (Sec. 2), producing filtered skeletons without needing any user input (Sec. 4). Then, we expose a new method, based on such skeletons, to enhance the results (see Fig 1.c and d) and the performance of the path tracing algorithm (Sec. 5).

## 2 The Cubical Complex Framework

In the 3D voxel framework, objects are made of voxels. In the 3D cubical complex framework, objects are made of cubes, squares, lines and vertices. Let  $\mathbb{Z}$  be the set of integers, we consider the family of sets  $\mathbb{F}_0^1$  and  $\mathbb{F}_1^1$ , such that  $\mathbb{F}_0^1 = \{\{a\} \mid a \in \mathbb{Z}\}$



and  $\mathbb{F}_1^1 = \{\{a, a + 1\} \mid a \in \mathbb{Z}\}$ . Any subset  $f$  of  $\mathbb{Z}^n$  such that  $f$  is the Cartesian product of  $m$  elements of  $\mathbb{F}_1^1$  and  $(n - m)$  elements of  $\mathbb{F}_0^1 = \{0\}$  is called a face or an  $m$ -face of  $\mathbb{Z}^n$ ,  $m$  is the dimension of  $f$ , we write  $\dim(f) = m$ . A 0-face is called a *vertex*, a 1-face is an *edge*, a 2-face is a *square*, and a 3-face is a *cube*.

Given  $m \in \{0, \dots, n\}$ , we denote by  $\mathbb{F}_m^n$  the set composed of all  $m$ -faces in  $\mathbb{Z}^n$ . We denote by  $\mathbb{F}^n$  the set composed of all  $m$ -faces in  $\mathbb{Z}^n$ :  $\mathbb{F}^n = \bigcup_{m \in [0;n]} \mathbb{F}_m^n$ .

Let  $f \in \mathbb{F}^n$ . We set  $\hat{f} = \{g \in \mathbb{F}^n \mid g \subseteq f\}$ , and  $\hat{f}^* = \hat{f} \setminus \{f\}$ . Any element of  $\hat{f}$  (resp.  $\hat{f}^*$ ) is a *face of  $f$*  (resp. a *proper face of  $f$* ). The *closure* of a set of faces  $X$  is the set  $\widehat{X} = \cup\{\hat{f} \mid f \in X\}$ .

**Definition 1.** A finite set  $X$  of faces in  $\mathbb{F}^n$  is a complex if  $X = \widehat{X}$ , and we write  $X \preceq \mathbb{F}^n$ .

Any subset  $Y$  of  $X$  which is also a complex is a subcomplex of  $X$ , and we write  $Y \preceq X$ .

A face  $f \in X$  is a *facet of  $X$*  if  $f$  is not a proper face of any face of  $X$ . The *dimension of  $X$*  is  $\dim(X) = \max\{\dim(f) \mid f \in X\}$ . If  $\dim(X) = d$ , then we say that  $X$  is a  $d$ -complex.

Traditionally, a binary image is a finite subset of  $\mathbb{Z}^n$  (called voxel image when  $n = 3$ ). To transpose such an image  $S$  to the cubical complex framework, we associate to each element of  $S \subseteq \mathbb{Z}^n$  an  $n$ -face of  $\mathbb{F}^n$ . Let  $x = (x_1, \dots, x_n) \in S$ , we define the  $n$ -face  $\Psi(x) = \{x_1, x_1 + 1\} \times \dots \times \{x_n, x_n + 1\}$ . We can extend the map  $\Psi$  to sets:  $\widehat{\Psi(S)} = \{\widehat{\Psi(x)} \mid x \in S\}$ . Given a set  $S \subset \mathbb{Z}^n$ , we associate to it the cubical complex  $\widehat{\Psi(S)}$ .

The collapse operation is the basic operation for performing homotopic thinning of a complex, and consists of removing two distinct faces  $(f, g)$  from a complex  $X$  under the condition that they form a free pair:

**Definition 2.** Let  $X \preceq \mathbb{F}^n$ , and let  $f, g$  be two faces of  $X$ . The face  $g$  is free for  $X$ , and the pair  $(f, g)$  is a free pair for  $X$  if  $f$  is the only face of  $X$  which strictly contains  $g$ .

It can be easily seen that if  $(f, g)$  is a free pair for a complex  $X$ , then  $f$  is a facet of  $X$  and  $\dim(g) = \dim(f) - 1$ .

**Definition 3.** Let  $X \preceq \mathbb{F}^n$ , and let  $(f, g)$  be a free pair for  $X$ . The complex  $X \setminus \{f, g\}$  is an elementary collapse of  $X$ .

Let  $Y \preceq \mathbb{F}^n$ . The complex  $X$  collapses onto  $Y$  if there exists a sequence of complexes  $(X_0, \dots, X_\ell)$  of  $\mathbb{F}^n$  such that  $X = X_0$ ,  $Y = X_\ell$  and for all  $i \in \{1, \dots, \ell\}$ ,  $X_i$  is an elementary collapse of  $X_{i-1}$ . We also say, in this case, that  $Y$  is a collapse of  $X$ .

Recent works in the cubical complex framework brought new parallel thinning algorithms in the voxel framework ([2], [1]).

### 3 A Parallel Directional Thinning Based on Cubical Complex

In the cubical complex framework, parallel removal of free pairs can be easily achieved when following simple rules that we give now. First, we need to define the *direction* and the *orientation* of a free face. Let  $(f, g)$  be a free pair for  $X \preceq \mathbb{F}^n$ : we have  $\dim(g) = \dim(f) - 1$ , and it can be seen that  $g = f \cap f'$ , where  $f'$  is the translate of  $f$  by one of the  $2n$  vectors of  $\mathbb{Z}^n$  which have all their coordinates equal to 0 except one, which is either equal to  $+1$  or  $-1$ . Let  $v$  be this vector, and  $c$  its non-null coordinate. We define  $Dir(f, g)$ , called the *direction* of the free pair  $(f, g)$ , as the index of  $c$  in  $v$ . The *orientation* of the free pair  $(f, g)$  is defined as  $Orient(f, g) = 1$  if  $c = 1$ , and  $Orient(f, g) = 0$  else.

Now, we give a property of collapse (previously proven in [4]) which brings a necessary and sufficient condition for removing two free pairs of faces in parallel from a complex, while preserving topology.

**Proposition 4.** *Let  $X \preceq \mathbb{F}^n$ , and let  $(f, g)$  and  $(k, \ell)$  be two distinct free pairs for  $X$ . The complex  $X$  collapses onto  $X \setminus \{f, g, k, \ell\}$  if and only if  $f \neq k$ .*

From Prop. 4, the following corollary is immediate.

**Corollary 5.** *Let  $X \preceq \mathbb{F}^n$ , and let  $(f_1, g_1) \dots (f_m, g_m)$  be  $m$  distinct free pairs for  $X$  such that, for all  $a, b \in \{1, \dots, m\}$  (with  $a \neq b$ ),  $f_a \neq f_b$ . The complex  $X$  collapses onto  $X \setminus \{f_1, g_1 \dots f_m, g_m\}$ .*

Considering two distinct free pairs  $(f, g)$  and  $(i, j)$  for  $X \preceq \mathbb{F}^n$  such that  $Dir(f, g) = Dir(i, j)$  and  $Orient(f, g) = Orient(i, j)$ , we have  $f \neq i$ . From this observation and Cor. 5, we deduce the following property.

**Corollary 6.** *Let  $X \preceq \mathbb{F}^n$ , and let  $(f_1, g_1) \dots (f_m, g_m)$  be  $m$  distinct free pairs for  $X$  having all the same direction and the same orientation. The complex  $X$  collapses onto  $X \setminus \{f_1, g_1 \dots f_m, g_m\}$ .*

Intuitively, we want our thinning algorithm to remove free faces of a complex “layer by layer” and to avoid having unequal thinning of the input complex. Therefore, we want each execution of the algorithm to remove free faces located on the border of the input complex. We define  $Border(X)$  as the set all faces belonging to a free pair for  $X$ . We now introduce Alg. 1, a directional parallel thinning algorithm.

On a single execution of the main loop of Alg. 1, only faces located on the border of the complex are removed (l. 7). Thanks to corollary 6, we can remove faces with same direction and orientation in parallel (l. 8), while guaranteeing topology preservation. Figure 2 depicts the first steps of the algorithm.

Different definitions of orientation and direction can be given, each corresponding to a different order of free faces removal in the complex and leading to different results. Algorithm 1 can be implemented to run in linear time complexity (proportionally to the number of faces in the complex). Indeed, checking if a face is free or not may be easily done in constant time and when a free pair  $(f, g)$  is removed from the input complex, it is sufficient to scan the faces contained in  $f$  in order to find new free faces.

---

**Algorithm 1.** *ParDirCollapse*( $X, W, \ell$ )

---

**Data:** A cubical complex  $X \preceq \mathbb{F}^n$ , a subcomplex  $W \preceq X$  which represents faces of  $X$  which should not be removed, and  $\ell \in \mathbb{N}$ , the number of layers of free faces which should be removed from  $X$

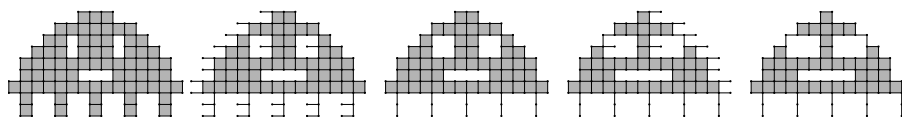
**Result:** A cubical complex

```

1 while there exists free faces in  $X \setminus W$  and  $\ell > 0$  do
2      $L = \widehat{\text{Border}}(X)$ ;
3     for  $t = 1 \rightarrow n$  do
4         for  $s = 0 \rightarrow 1$  do
5             for  $d = n \rightarrow 1$  do
6                  $E = \{(f, g) \text{ free for } X \mid g \notin W, \text{Dir}(f, g) = t, \text{Orient}(f, g) = s,$ 
7                      $\dim(f) = d\}$ ;
8                  $G = \{(f, g) \in E \mid f \in L \text{ and } g \in L\}$ ;
9                  $X = X \setminus G$ ;
9              $\ell = \ell - 1$ ;
10 return  $X$ ;

```

---



**Fig. 2.** Four first iterations of Alg. 1 running on the left-most shape

## 4 Aspect Preservation during Thinning

As previously said, our goal is to use the skeleton of the voids of a scene in order to guide light rays from the camera of the scene towards the light. Moreover, this skeleton needs to capture the main geometrical features of the original scene. For example, if the input is a corridor (the void of the corridor), then the skeleton should be a line following the main direction of the corridor.

Generally, two strategies are possible to achieve this goal: find, during the skeletonization process, points whose neighbourhood configuration seems interesting and keep them in the result [6] [7] [5], or choose, before skeletonization, interesting points of the object which should remain untouched, based on a function on these points and a filtering parameter [2] [12].

Algorithm 1 does not necessarily preserve geometrical features of the input object in the resulting skeleton (for example, the skeleton of a corridor could be reduced to a single vertex). In the following, we introduce a new method in the cubical complex framework, requiring no user input, for obtaining a curvilinear skeleton yielding satisfactory geometrical properties. Our method is based on the two previously listed strategies: it finds, during thinning, elements with a specific neighbourhood configuration, and uses a function on these elements to decide whether to preserve them, or not, in the result.

### 4.1 The Lifespan of a Face

In the following, we define additional functions in the cubical complex, related to the thinning process (Alg. 1), which are essential for the filtering step of the skeletonization. The first one we present is the *death date* of a face.

**Definition 7.** *Let  $f \in X \preceq \mathbb{F}^n$ . The death date of  $f$  in  $X$ , denoted by  $Death_X(f)$ , is the smallest integer  $\delta$  such that  $f \notin ParDirCollapse(X, \emptyset, \delta)$ .*

Intuitively, the death date of a face indicates how many layers of free faces should be removed from a complex  $X$ , using Alg. 1, before removing completely the face from  $X$ . We now define the *birth date* of a face:

**Definition 8.** *Let  $f \in X \preceq \mathbb{F}^n$ . The birth date of  $f$  in  $X$ , denoted by  $Birth_X(f)$ , is the smallest integer  $b$  such that either  $f$  is a facet of  $ParDirCollapse(X, \emptyset, b)$ , or  $f \notin ParDirCollapse(X, \emptyset, b)$ .*

The birth date indicates how many layers of free faces must be removed from  $X$  with Alg.1 before transforming  $f$  into a facet of  $X$  (we consider that a face “lives” when it is a facet). Finally, we define the *lifespan* of a face :

**Definition 9.** *Let  $f \in X \preceq \mathbb{F}^n$ . The lifespan of  $f$  in  $X$  is the integer*

$$Lifespan_X(f) = \begin{cases} +\infty & \text{if } Death_X(f) = +\infty \\ Death_X(f) - Birth_X(f) & \text{otherwise} \end{cases}$$

These three values depend on the order of direction and orientation chosen for Alg. 1.

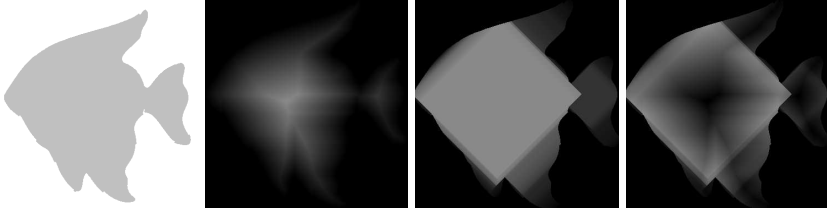
The lifespan of a face  $f$  of  $X$  indicates how many “rounds” this face “survives” as a facet in  $X$ , when removing free pairs with Alg. 1, and is a good indicator of how important a face can be in an object. Typically, the higher the lifespan is, and the more representative of an object’s geometrical feature the face is. The lifespan, sometimes called *saliency*, was used in [10] (with the name “medial persistence”) in order to propose a thinning algorithm in cubical complexes based on two parameters.

### 4.2 Distance Map, Opening Function and Decentness Map

In addition to the lifespan of a face, the proposed homotopic thinning method uses information on distance between faces in order to decide if a face should be kept safe from deletion. We define hereafter various notions based on distances in the voxel framework.

We set  $d_1(x, y)$  as the L1 distance between  $x$  and  $y$  (Manhattan distance). Let  $S \subset \mathbb{Z}^n$ , we set  $S^c = \mathbb{Z}^n \setminus S$ , and for all  $x \in \mathbb{Z}^n$ , the map  $D_1(S) : \mathbb{Z}^n \rightarrow \mathbb{N}$  is such that  $D_1(S)(x) = \min_{y \in S^c} d_1(x, y)$ .

The *maximal 1-ball in  $S$  centered on  $x$*  is the set  $\mathbb{M}_S^1(x) = \{y \in \mathbb{Z}^n \mid d_1(x, y) < D_1(S)(x)\}$ . We set, for all  $x \in S$ , the map  $\Omega_1(S) : \mathbb{Z}^n \rightarrow \mathbb{N}$  such that  $\Omega_1(S)(x) = \max_{y \in \mathbb{M}_S^1(x)} D_1(S)(y)$ : this value indicates the radius of a largest maximal 1-ball



**Fig. 3. Examples of opening and decenterness map** - From left to right: a shape  $S \subset \mathbb{Z}^2$  (in gray),  $D_1(S)$ ,  $\Omega_1(S)$  and  $\mathcal{DC}_1(S)$  (low values have dark colour)

contained in  $S$  and containing  $x$ . If  $x \in S^c$ , we set  $\Omega_1(S)(x) = 0$ . The map  $\Omega_1(S)$  is known as the opening function of  $S$  based on the 1-distance (also called the granulometry function) [11]: it allows to compute efficiently results of morphological openings by balls of various radius, and gives information on the local thickness of an object.

Given  $S \subset \mathbb{Z}^n$ , the value of  $\Omega_1(S)(x)$  of every  $x \in S$  can be naively computed by performing successive morphological dilations of values of the map  $D_1(S)$ . A linear algorithm for computing the map  $\Omega_1(S)$  (with regard to the size of the input image) was proposed in [3], and will be explored further in details in a future paper.

Finally, we define the *decenterness map*:

**Definition 10.** Given  $S \subset \mathbb{Z}^n$ , the decenterness map of  $S$  is the map  $\mathcal{DC}_1(S) = \Omega_1(S) - D_1(S)$ .

An example of these maps is shown on Fig. 3.

In order to extend all these previous maps defined in  $\mathbb{Z}^n$  to the cubical complex framework, we use the map  $\Psi^{-1}$ , inverse of the bijective map  $\Psi : \mathbb{Z}^n \rightarrow \mathbb{F}_n^n$  defined in Sec. 2. It is used to project any  $n$ -face of  $\mathbb{F}^n$  into  $\mathbb{Z}^n$ . This map induces a map from  $\mathcal{P}(\mathbb{F}_n^n)$  to  $\mathcal{P}(\mathbb{Z}^n)$ , that we also denote by  $\Psi^{-1}$ .

Given  $Y \subset \mathbb{F}^n$ , we set  $S = \Psi^{-1}(Y \cap \mathbb{F}_n^n)$ . We define the map  $D_1^{cc}(Y) : \mathbb{F}^n \rightarrow \mathbb{N}$  as follows: for all  $f \in \mathbb{F}^n$ ,

$$D_1^{cc}(Y)(f) = \begin{cases} D_1(S)(\Psi^{-1}(f)) & \text{if } f \text{ is an } n\text{-face} \\ \max_{g \in (\hat{g}^* \cap \mathbb{F}_n^n)} D_1^{cc}(Y)(g) & \text{otherwise} \end{cases}$$

Informally, if  $f$  is a 3-face, then  $D_1^{cc}(Y)(f)$  is the length of the shortest 1-path between the voxel “corresponding” to  $f$  and the set of voxels corresponding to  $Y$ . In the same way, we define  $\Omega_1^{cc}(Y)$  and  $\mathcal{DC}_1^{cc}(Y)$ .

### 4.3 Parameter-Free Filtered Thinning

As previously said, we add edges to the set  $W$  of Alg. 1 in order to retain, in the resulting curvilinear skeleton, important edges from the original object. Given

**Algorithm 2.** *CurvilinearSkeleton*( $X$ )**Data:** A cubical complex  $X \preceq \mathbb{F}^3$ **Result:** A cubical complex  $Y \preceq \mathbb{F}^3$ 

- 1  $W = \{f \in X \mid Lifespan_X(f) > DC_1^{cc}(X)(f) + Birth_X(f) - D_1^{cc}(X)(f) \text{ and } \dim(f) = 1\}$  ;
- 2 **return** *ParDirCollapse*( $X, W, +\infty$ );

a cubical complex  $X$ , if an edge of  $X$  has a high decenterness value for  $X$ , then it is probably located too close to the border of  $X$  and does not represent an interesting geometrical feature to preserve. On the other hand, if an edge has a high lifespan for  $X$ , then it means it was not removed quickly, after becoming a facet, by the thinning algorithm and might represent some precious geometrical information on the original object. An idea would be to keep, during thinning, all edges whose lifespan is superior to the decenterness value. Unfortunately, this strategy produces skeletons with many spurious branches in surfacic areas of the original object.

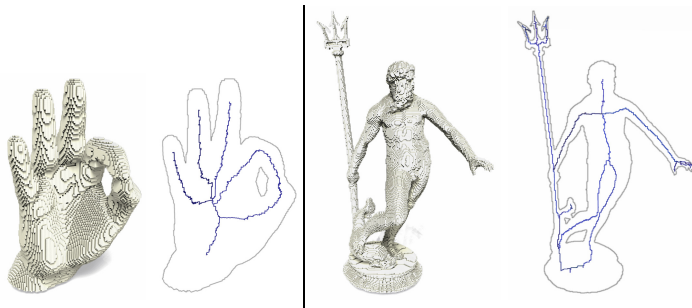
We can identify surfacic areas of a complex as zones where squares have a high lifespan. Therefore, in order to avoid spurious branches in surfacic areas, we need to make it harder for edges to be preserved in these zones. It can be achieved by deciding that an edge will be kept safe from deletion by the thinning algorithm if its lifespan is superior to the decenterness value plus the lifespan of squares “around” this edge. This leads us to proposing Alg. 2.

In order to understand what was realised on line 1 of Alg. 2, we might point out that the birth date of an edge corresponds to the highest death date of the squares containing this edge. Moreover, the map  $D_1^{cc}(X)$  gives, for all 3-faces of  $X$ , their death date (as the thinning algorithm naturally follows this map to eliminate cubes from a 3-complex). Therefore, for an edge  $f$  of  $X$ ,  $D_1^{cc}(X)(f)$  informs us on the highest death date of cubes containing  $f$ , also equal to the highest birth date of squares containing  $f$ . In conclusion,  $Birth_X(f) - D_1^{cc}(X)(f)$  is an approximation of the lifespan of the squares containing  $f$ .

Although the output of Alg.2 may contain 2-faces, the algorithm is said to be a *curvilinear skeletonization* algorithm because it only adds 1-faces (edges) in the inhibitor set  $W$  (used for the visual aspect preservation step) (on line 1 of Alg. 2). For the same reason, we say that the output of Alg. 2 is a *curvilinear skeleton*. In the scenes studied in Sec. 5, the outputs of Alg. 2 were one dimensional complexes.

#### 4.4 Results

Algorithm 2 allows to obtain a filtered curvilinear skeleton from a three dimensional complex. The results presented in Fig. 4 show that the skeletons contain the main geometrical information from the input shapes, and no spurious branch.



**Fig. 4. Results of algorithm 2** for two shapes: a hand (left) and a statue (right). In each pair, the rightmost image represents the skeleton.

## 5 Application to Path Tracing

In this section, we explain how the curvilinear skeleton of the voids of the scene can enhance the path tracing algorithm. We start with some basic elements related to path tracing and then present our modified path tracing algorithm.

### 5.1 The Path Tracing

Let  $O$  be the origin of the camera in the scene. For each pixel  $P$  of the final image, let  $x$  be the nearest intersection point of the ray  $(O, \vec{OP} = -\Theta)$  with an element of the scene. To obtain the luminosity, we must solve the rendering equation [8] which is a recursive integral equation, the integrand containing the radiance function that we must compute:

$$L(x \rightarrow \Theta) = L_e(x \rightarrow \Theta) + \int_{\Phi \in \Omega_x} f_s(x, \Theta \leftrightarrow \Phi) L(r(x, \Phi) \rightarrow -\Phi) d\omega_\Phi$$

where  $L$  is the radiance (a radiometric quantity that represents luminosity) from point  $x$  toward direction  $\Theta$  and  $L_e$  is the emitted radiance (non null only on light sources). The point  $r(x, \Phi)$  is the nearest visible point from  $x$  in the direction  $\Phi$  of the hemisphere  $\Omega_x$ . The  $f_s$  function expresses how much luminosity is exchanged, on point  $x$ , between an incoming ray  $\Phi$  and the outgoing ray  $\Theta$ .

The previous equation expresses an intuitive idea: the reflected radiance from  $x$  towards the camera is the result of computing all incoming luminosity on  $x$  scaled by a factor which depends on the material on  $x$  and the angle of the ray going from the camera to  $x$ . The most common method used to estimate the integral, denoted by  $L_r(x \rightarrow \Theta)$ , is the Monte-Carlo integration which provides the following:

$$\langle L_r(x \rightarrow \Theta) \rangle = \frac{\int f_s(x, \Theta \leftrightarrow \Phi) L(r(x, \Phi) \rightarrow -\Phi) p(\Phi) d\omega_\Phi}{p(\Theta)}$$

The function  $p$  is a probability density function (pdf) that is used to sample  $\Phi$ . This estimator is unbiased, meaning that the expected value  $E[\langle L_r(x \rightarrow \Theta) \rangle]$

is equal to  $L_r(x \rightarrow \Theta)$ . The variance of the estimator expresses its quality and depends on the chosen pdf  $p$ . The best choice is a pdf that matches the shape of the function to integrate (ie. gives high density to samples that have high values for the function and low density to samples that have low values). The strategy of choosing an adapted pdf is called *importance sampling* [8] and is used in global illumination to improve the convergence speed of the algorithms.

The algorithm computes  $L$  as follow: it chooses one random direction  $\Phi$  based on the pdf  $p$  and applies the estimator by calling  $L$  recursively to compute  $L_r$  (shoots a new ray in the direction  $\Phi$ , computes the new hit point with the scene, and compute the new radiance at this point). Finally, it returns the sum of the emitted radiance  $L_e$  and the result for  $L_r$ . The recursion stops when either a maximal number of bounces or a directly illuminated area have been reached.

This computation is done multiple times for each pixel. We average the results and get an estimation of the mean radiance passing through the pixel and heading towards the camera. A bad pdf would lead in picking directions that do not reach the light before the end of the recursion, and produce results with a lot of noise in the final image. In the next part, we explain how to produce a pdf based on the curvilinear skeleton.

## 5.2 Skeleton Based Importance Sampling

As stated in the introduction, a curvilinear skeleton of the void (with some preprocessing performed on it) gives us information on which directions the light comes from. Given these directions, we can build a efficient pdf  $p_{skel}$  and guide our rays by sampling the hemispheres with  $p_{skel}$ . The integrand of  $L_r$  is:

$$f_s(x, \Theta \leftrightarrow \Phi)L(r(x, \Phi) \rightarrow -\Phi).$$

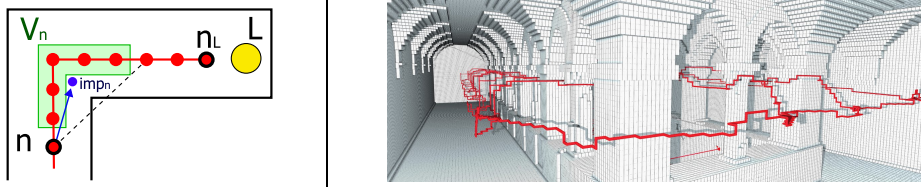
The most common strategy used to sample  $\Omega_x$  is to use the function  $f_s$  because it is an input of the algorithm. The term  $L$ , representing the distribution of light in the scene, is unknown. Our method gives a way to sample  $L$ .

**Construction of the Importance Points.** The skeleton of the voids of the scene is computed using Alg. 2 (as shown on Fig. 5 on the right) and is converted to a graph (the nodes are the 0-faces and the edges are the 1-faces). We then compute a set of *importance points*, which will be used to sample  $\Omega_x$  in the path tracing algorithm. To each node  $n$  of the skeleton, one importance point  $imp_n$  is computed. Intuitively, the importance point associated to  $n$  is the direction to follow in order to find a light source.

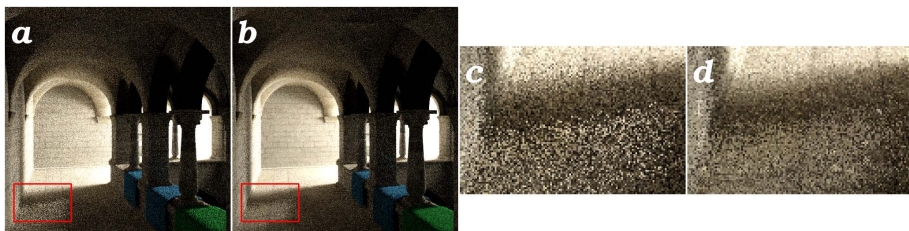
Let  $L$  be the light source of the scene and  $n_L$  the nearest node of the skeleton that is visible by  $L$ . For each node, we compute the shortest path to  $n_L$  along the skeleton. To do so, we use the Dijkstra algorithm and weight the edges of the skeleton depending on a visibility criteria: the weight of an edge  $e$  is 1 if  $e$  is visible from  $n_L$  and 10 else. It results that illuminated paths will be shorter than paths located in dark areas.

Let  $n$  be a node of the skeleton and  $V_n$  the set of visible nodes from  $n$  along the shortest path toward  $n_L$ . The importance point  $imp_n$  associated to  $n$  is the





**Fig. 5.** On the left, we illustrate the importance point  $imp_n$  associated to a node  $n$ . The dotted black line shows the closest node to  $n$  not included in  $V_n$ . On the right, an example of the curvilinear skeleton (in red) obtained in the Sponza scene.



**Fig. 6.** Same scene rendered using the classical path tracing (image  $a$ ) or our method (image  $b$ ). Images  $c$  and  $d$  are both details of respectively images  $a$  and  $b$

Scene	Corridor	Sponza 1	Sponza 2 (Fig. 6)
MSE 100	145 / 57	301 / 156	881 / 826
MSE 40	434 / 260	924 / 628	2678 / 2570

**Fig. 7.** Time (s) to reach an MSE of 100 and 40 against reference images. In each cell, the left number is for the standard path tracing, the right is for our method

barycenter of  $V_n$ . An example is shown on Fig. 5 on the left. The algorithm can be extended to multiple light sources by taking into account, when computing the importance point of a skeleton node, only the closest light source.

**Sampling According to  $L$ .** Given a point  $x$  on the scene and a direction  $\Theta$ , we want to compute  $L(x \rightarrow \Theta)$  and then sample the hemisphere  $\Omega_x$ . We search for the nearest skeleton node  $n$  to  $x$  and its importance point  $imp_n$ . We sample the hemisphere with a power-cosine pdf centered on  $\overrightarrow{x imp_n}$ :

$$p_{skel}(\Phi) = \frac{s+1}{2\pi} * \cos^s \alpha$$

with  $\alpha$  the angle between  $\overrightarrow{x imp_n}$  and  $\overrightarrow{\Phi}$ ,  $s$  being a parameter called skeleton strength. The higher  $s$  is, the closer to  $\overrightarrow{x imp_n}$  we sample.

### 5.3 Results and Discussion

Some results are presented on Fig. 6 and Fig. 1 where we can see that our method produces less noisy images compared to the regular path tracing. We present, in Fig 7, the time taken by each method to produce an image of same quality (the quality is measured by the mean square error (MSE) with a reference image) in different scenes: our method is the fastest.

## 6 Conclusion

We presented in this article a new skeletonization algorithm that both preserves geometrical features of objects and produces a pure curvilinear skeleton. These two properties allow us to improve the path tracing algorithm in guiding the rays towards the main illuminated area of the scene. Our algorithm is faster and produce less noise than the classical path tracing method.

## References

1. Bertrand, G.: On critical kernels. *Comptes Rendus de l'Académie des Sciences, Série Mathématiques I*(345), 363–367 (2007)
2. 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)
3. Chaussard, J.: Topological tools for discrete shape analysis. Ph.D. thesis, Université Paris-Est (December 2010)
4. 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)
5. Chaussard, J., Couprie, M., Talbot, H.: Robust skeletonization using the discrete lambda-medial axis. *Pattern Recognition Letters* (2010) (in press)
6. Couprie, M., Coeurjolly, D., Zrour, R.: Discrete bisector function and euclidean skeleton in 2D and 3D. *Image and Vision Computing* 25(10), 1543–1556 (2007)
7. Hesselink, W.H., Roerdink, J.B.T.M.: Euclidean skeletons of digital image and volume data in linear time by the integer medial axis transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30(12), 2204–2217 (2008)
8. Kajiya, T.: The Rendering Equation. *Computer Graphics (ACM SIGGRAPH 1986 Proceedings)* 20(4), 143–150 (1986)
9. Lafortune, E.P., Willems, Y.D.: Bi-directional path tracing. In: *Proceedings of the 3rd International Conference on Computational Graphics and Visualization Techniques*, pp. 145–153 (1993)
10. Liu, L., Chambers, E.W., Letscher, D., Ju, T.: A simple and robust thinning algorithm on cell complexes. *Computer Graphics Forum (Proceedings of Pacific Graphics 2010)* (2010)
11. Matheron, G.: *Eléments pour une Théorie des Milieux Poreux* (1967)
12. Palágyi, K.: A Subiteration-Based Surface-Thinning Algorithm with a Period of Three. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) *DAGM 2007*. LNCS, vol. 4713, pp. 294–303. Springer, Heidelberg (2007)

# From the Zones of Influence of Skeleton Branch Points to Meaningful Object Parts

L. Serino, C. Arcelli, and G. Sanniti di Baja

Institute of Cybernetics “E. Caianiello”, CNR, Naples, Italy  
{l.serino,c.arcelli,g.sannitidibaja}@cib.na.cnr.it

**Abstract.** A 3D object decomposition method is presented, which is based on the decomposition of the linear skeleton guided by the zones of influence. These are the connected components of voxels obtained by applying the reverse distance transformation to the branch points of the skeleton. Their role is to group sufficiently close branch points and to detect perceptually meaningful skeleton branches that are in a one-to-one relation with the object parts.

## 1 Introduction

According to the structural approach to shape analysis, an object can be interpreted as constituted by a number of perceptually meaningful parts and its description can be given in terms of the description of the various parts and of their spatial relationships. This approach has been inspired by the behavior of the human visual system, as discussed in [1-4]. One of the advantages of such a structured representation is a greater robustness under changes in viewing conditions.

The skeleton is a tool often employed to achieve a structural analysis of the object it represents [5-12]. In fact, the skeleton is a linear subset of the object reflecting the topological and geometrical features of the object and such that each skeleton branch is in correspondence with one of the parts understood as constituting the object. Thus, a decomposition of an object into its constituting parts can be guided by a decomposition of the skeleton into its constituting branches.

Parts associated with skeleton branches meeting in common points, the branch points of the skeleton, overlap with each other. If decomposition into disjoint parts is preferred, care is necessary to deal with the overlapping regions.

We have suggested shape decomposition methods guided by skeleton decomposition for both 2D and 3D objects. In the 2D case, we decomposed the skeleton into its constituting branches and obtained object decomposition into partially overlapping parts [5,6]. In the 3D case, we favored decomposition into disjoint parts and to this purpose we suggested a suitable partition of the skeleton [10,11]. In particular in [11], we partitioned the skeleton into three types of components, respectively called complex sets, simple curves and single points, which correspond to three types of object parts, respectively called kernels, simple regions and bumps. Simple regions and bumps protrude from the kernels. In turn, kernels can

be interpreted as sort of main bodies of the object. Kernels were identified in correspondence with the positions where different skeleton branches meet.

In this paper, we work with 3D objects and propose a decomposition method that is inspired by our work in 2D as concerns the decomposition of the skeleton into its constituting branches, and shares with our 3D method the fact that the object is decomposed into disjoint parts. The current method will face the problem of identifying skeleton branches corresponding to meaningful object parts, and will deal with the assignment of each overlapping region to only one of the proper object parts overlapping each other.

The object parts obtained as described in this paper are somehow analogous to the simple regions and bumps of the decomposition method in [11]. In both cases, object decomposition is into disjoint parts. An important difference between the two methods is that with the current method no region merging is necessary to achieve an object decomposition in accordance with human intuition. Other analogies and differences will be discussed in Section 4.

## 2 Preliminaries

We consider objects rid of cavities in binary voxel images in cubic grids. The 26-connectedness is used for the object and the 6-connectedness for the background. The neighbors of a voxel  $p$  are the 26 voxels sharing with  $p$  a face, an edge, or a vertex.

The distance between two voxels  $p$  and  $q$  is defined as the length of a minimal discrete path linking  $p$  to  $q$ . The three integer weights  $w_f=3$ ,  $w_e=4$  and  $w_v=5$  are used to measure the unit moves from a voxel towards its face-, edge- and vertex-neighbors along the path, respectively. This choice of weights is motivated by the fact that the so obtained  $\langle 3,4,5 \rangle$  weighted distance provides a reasonably good approximation to the Euclidean distance [13].

According to the model proposed by Blum [14], the skeleton of an object is a subset of the object consisting of points symmetrically placed within the object, having the same topology of the object, and such that each skeleton point is associated with its distance from the background. The value of a skeleton point can be interpreted as the radius of a ball that, centered on the point, is bi-tangent to the object's boundary and is included in the object. The object can be recovered starting from its skeleton by computing the envelope of the balls associated to its points.

For 3D objects, the above model originates a surface skeleton, which consists of the union of surfaces and curves. The surface skeleton of objects rid of cavities can be furthermore reduced to originate a skeleton exclusively consisting of curves. Only partial object recovery from such a skeleton is possible, unless the object consists of parts with tubular shape. In fact, only in such a case the symmetry points are mostly aligned along symmetry axes, while in the general case symmetry points are placed along symmetry planes and axes.

In the digital space, the skeleton of a 3D object can be computed according to the model of Blum by identifying the centers of maximal balls in the distance transform of the object. The distance transform DT is a multi-valued replica of the object, where

each voxel is labeled with its distance from the background. Thus, each voxel in DT can be interpreted as the center of a ball with radius equal to the corresponding distance label. In particular, a voxel whose associated ball is not included by any other single ball in the object is called center of maximal ball CMB. The object can be recovered by the union of the balls associated to the CMBs of the object. The ball associated to any distance labeled voxel  $p$  can be obtained by applying to  $p$  the reverse distance transformation [15].

In this paper, we compute DT by using the  $\langle 3,4,5 \rangle$  weighted distance. There, to establish whether a voxel  $p$  is a CMB it is enough to compare the distance label of  $p$  with the distance labels of its 26 neighbors, by suitably taking into account the weights  $w_f$ ,  $w_e$  and  $w_v$  [16]. As for the skeleton used to guide object decomposition, we refer to the linear skeleton obtained by the algorithm for DT based skeletonization suggested in [17]. We are aware that when using a linear skeleton a difference generally exists between the input object and the union of the only balls associated with the voxels of its skeleton. Thus, our decomposition method is completely effective only in case of objects that are perceived as consisting of parts with tubular shape, since in this case the above difference is negligible. This is the object domain considered in the following.

### 3 The Method

Let  $S$  be the skeleton of the object at hand. A voxel  $p$  of  $S$  is an end point when it has only one neighbor in  $S$ , is a normal point when it has two neighbors in  $S$ , and is a branch point when it has more than two neighbors in  $S$ . A skeleton branch is a curve of  $S$  entirely consisting of normal points, except for the two extremes of the curve that are end points or branch points.

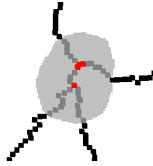
Balls associated to a set of distance labeled skeleton voxels by the reverse distance transformation may overlap and merge into connected components. Let us consider the balls associated with all the branch points of  $S$ . Each group of these balls forming a connected component is called *zone of influence* of the branch points it includes. Branch points that are neighbors of each other or are closer to each other than the sum of their associated distance labels are included in the same zone of influence.

Intersecting object parts are mapped into branches of an ideal continuous skeleton that meet in a single branch point. In turn, more than one single branch point is generally found in the digital skeleton  $S$  in correspondence with intersecting object parts. However, in any such a case the branch points are likely to be very close to each other, so that they are included in a unique zone of influence. Thus, the zones of influence can be used to group branch points of  $S$  actually corresponding to a single branch point configuration of the skeleton that would ideally represent the object at hand. Obviously, the number of zones of influence may be smaller than the number of branch points of  $S$ .

In the following, the zones of influence are used to correctly identify the configurations where skeleton branches meet with each other. They are also used to count the number of perceptually meaningful branches of the skeleton and, hence, the

number of object decomposition parts. To this aim, connected component labeling is accomplished on the zones of influence, as well as on the set that is obtained from  $S$  by removing from it all voxels that result to be included in any zone of influence.

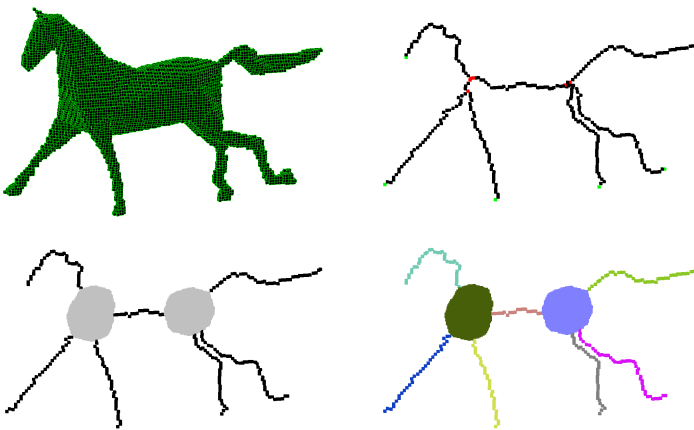
Short branches may exist in  $S$ , whose voxels are all included in a zone of influence, e.g., see Fig. 1. These short branches do not correspond to meaningful object parts and, hence, should not be counted. This is guaranteed since connected component labeling of  $S$  is done after removal of the skeleton voxels placed in zones of influence.



**Fig. 1.** A zone of influence (gray) including a short skeleton branch

### 3.1 Detecting Skeleton Components

To illustrate our object decomposition method, let us refer to the object “horse”, shown in Fig. 2 top left, which will be used as running example. The skeleton  $S$  computed by the algorithm [17] is shown in Fig. 2 top right, where red, black and green are used for branch points, normal points and end points, respectively. We note that  $S$  includes six branch points, six end points, and nine skeleton branches. The two zones of influence resulting after applying the reverse distance transformation to the six branch points of  $S$  are shown in Fig. 2 bottom left; finally, the result obtained by applying connected component labeling to the zones of influence and to the voxels outside them is shown in Fig. 2 bottom right, where different colors represent different identity labels.



**Fig. 2.** From top left to bottom right: the object “horse”, its skeleton, the zones of influence and the result of connected component labeling

Still with reference to Fig. 2, we note that out of the nine skeleton branches initially detected in the skeleton  $S$ , only seven branches having at least one voxel outside the zones of influence are identified as having perceptual significance. These skeleton branches correspond to the object parts of which the object can be interpreted as constituted. The seven object parts intersect with each other in correspondence of the zones of influence. However, as it will be explained in the following, the actual overlapping region among the object parts in correspondence of a given zone of influence is likely to be larger than the zone of influence itself.

### 3.2 Identifying Object Decomposition Components

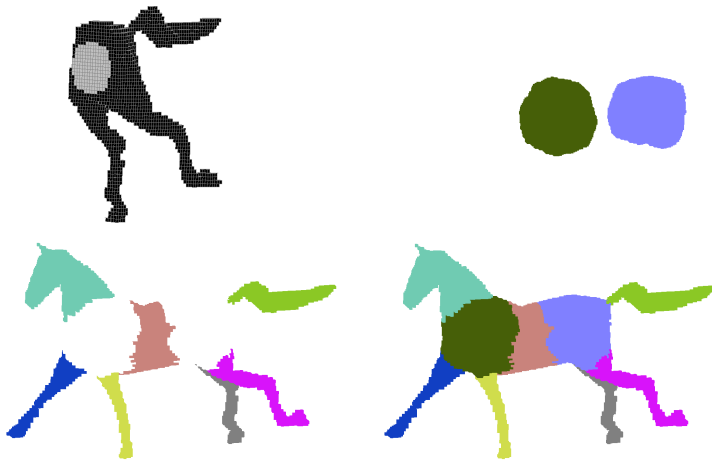
Ideally, by subtracting from the input object the zones of influence we should obtain a number of connected components of object voxels equal to the number of skeleton branches counted by connected component labeling. However, notwithstanding the fact that the image domain considered in this work includes only objects perceived as consisting of parts with almost tubular shape, a difference unavoidably exists between the volume of the input object and the volume of the object that could be obtained by applying to  $S$  the reverse distance transformation. Each zone of influence is certainly adjacent somewhere to the original background, but such an adjacency does not regard the whole surface delimiting the zone of influence. Thus, the number of achieved connected components of object voxels is generally smaller than expected. See Fig. 3 top left, where a section (black voxels) of the set that would be obtained by subtracting from “horse” the zones of influence (gray voxels) is shown. We observe that only one connected component of object voxels would remain after subtraction, while our aim is to decompose that part of the object into the four regions (torso, tail and the two back legs) that are in correspondence with the four detected perceptually meaningful skeleton branches.

To solve the above problem, we need to expand the zones of influence, so as to identify the proper overlapping regions. We aim at overlapping regions such that the cuts resulting in the object, when subtracting the overlapping regions from the input object, are in correspondence with significant curvature changes along the boundary of the object. To avoid both a too little expansion that would not produce the desired separation of the object parts, and an excessive expansion that would originate parts with unnatural separation cuts, we exploit distance information available in DT. In practice, the voxels of the zones of influence are labeled with the distances pertaining to them in DT. Then, the expansion is achieved by applying the reverse distance transformation to the so obtained distance labeled zones of influence.

The above process guarantees that the surfaces of the overlapping regions have a high adjacency degree with the original background. At the same time, reverse distance transformation also guarantees that unnatural cuts are not produced when the overlapping regions are subtracted from the input object. In fact, the zones of influence are nearly convex regions and this geometric property is preserved when the zones of influence are expanded. Thus, the overlapping regions cannot extend beyond the curvature minima along the boundary of the object. The overlapping regions obtained for the running example are shown in Fig. 3 top right.

We remark that the number of object parts has already been determined by counting the number of connected components of skeleton voxels outside the zones of influence. Thus, care should be taken to avoid diminishing such a number due to a possible fusion of the regions obtained when applying the reverse distance transformation to the distance labeled zones of influence. To this aim, a topology preserving reverse distance transformation is taken into account so as to avoid fusion. In practice, topology is maintained by setting to the background value the voxels that, though reached by the expansion of the zones of influence, result to be at the same distance from more than one zone of influence.

The overlapping regions are subtracted from the input object and the identity labels assigned to the voxels of the perceptually meaningful skeleton branches are finally used to label the connected components of object voxels they belong to. See Fig.3 bottom left, where the components of object voxels are colored as the corresponding perceptually meaningful skeleton branches. In this way, the preliminary object decomposition shown in Fig. 3 bottom right is obtained, where each overlapping region, colored as the influence zone from which has been obtained, still has to be ascribed to the proper component of object voxels.



**Fig. 3.** From top left to bottom right: section showing in gray the voxels of the zones of influence; the overlapping regions obtained by applying the topology preserving reverse distance transformation to the zones of influence; the components of object voxels obtained by subtracting from the object the overlapping regions; and the preliminary decomposition of the object before ascribing the overlapping regions to the proper object parts

For the sake of completeness, we point out that when computing the set difference between the input object and the overlapping regions, some connected components composed by a small number of object voxels may exist, which do not include any skeletal voxel. Such components are likely to exist since their voxels were not reached when applying the reverse distance transformation to the zones of influence due to the fact that  $S$  does not include all the CMBs of the object. These components



do not correspond to meaningful object parts and their voxels are assigned to the overlapping regions they are adjacent to.

### 3.3 Ascribing the Overlapping Regions

According to our model, the object should be decomposed into a number of disjoint parts equal to the number of perceptually meaningful skeleton branches. To reach this goal, a decision must be taken to ascribe each overlapping region to only one of the disjoint components of object voxels adjacent to it.

By observing an overlapping region, say  $OR_k$ , and the adjacent components of object voxels, say  $P_1, P_2, \dots, P_n$ , we may note that while  $OR_k$  is an almost convex set,  $P_1, P_2, \dots, P_n$  show a different degree of concavity in correspondence with the positions where they result to be adjacent to  $OR_k$ . Let  $A_k$  denote the area of the surface bounding the overlapping region  $OR_k$ , i.e., the number of voxels of  $OR_k$  having at least one face-neighbor outside  $OR_k$ . Moreover, let  $A_k(P_i)$  denote the portion of the area of the surface bounding the overlapping region  $OR_k$  that is adjacent to  $P_i$ , i.e., the number of voxels of  $OR_k$  having at least one face-neighbor in  $P_i$ . Then, we roughly evaluate how much  $OR_k$  intrudes into the adjacent component  $P_i$ , by computing the ratio  $R = A_k(P_i)/A_k$ . We ascribe the overlapping region  $OR_k$  to the adjacent component of object voxels that maximizes the ratio  $R$ . The choice of this criterion is due to the fact that in our opinion the more  $OR_k$  intrudes in a given component  $P_i$ , the more the shape of  $P_i$  benefits if the overlapping region is ascribed to it.

To accomplish the assignment of the overlapping regions in a computationally convenient manner, an adjacency matrix is built having as many rows as many are the overlapping regions and a number of columns equal to the number of components of object voxels plus one for the background. By inspecting the array where the preliminary decomposition of the object is stored, each time that a voxel of  $OR_k$  is met having at least one face neighbor outside  $OR_k$ , the proper element of the  $k$ -th row of the matrix is increased by one. In this way, once the matrix has been built, the value at row  $k$  and column  $j$  measures the portion of surface of  $OR_k$  in common with the background or with one of the adjacent components of object voxels. Then, we can easily decide to which component of object voxels to assign each overlapping region. In the rare case in which for an overlapping region an identical ratio is obtained for more than one adjacent component, the overlapping region is assigned to any of such components.

The adjacency matrix for the running example is shown in Table 1, where the two overlapping regions  $OR_1$  and  $OR_2$  are respectively those colored in green and blue in Fig. 3 bottom right.

**Table 1.** The adjacency matrix for the running example

	background	leg1	leg2	leg3	leg4	neck	torso	tail
$OR_1$	1517	44	33	0	0	125	411	0
$OR_2$	1622	0	0	70	53	0	414	5

The decomposition obtained for the running example is shown in Fig. 4 from two different view points.



Fig. 4. The obtained decomposition seen from two different view points

## 4 Experimental Results and Discussion

We have tested our method on 3D objects taken from publicly available shape repositories, e.g., [18], as done by most of the researchers in this field. The obtained results are generally satisfactory. In particular, for some objects used also in [7, 9, 12] to show the performance of the corresponding decomposition methods, our results seem to be qualitatively better. In Fig. 5 a few examples are given to show the performance of our decomposition method.

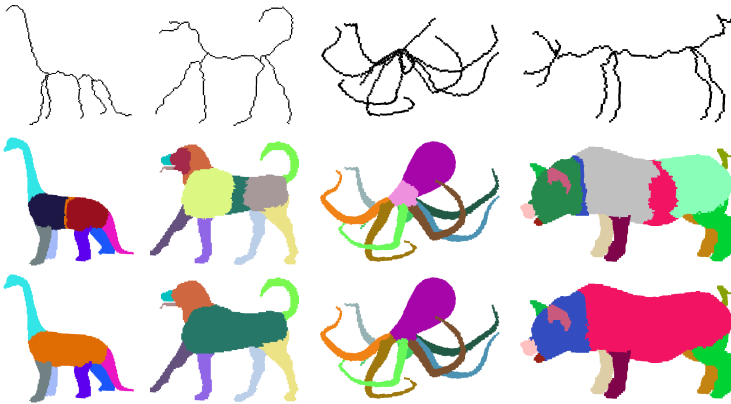
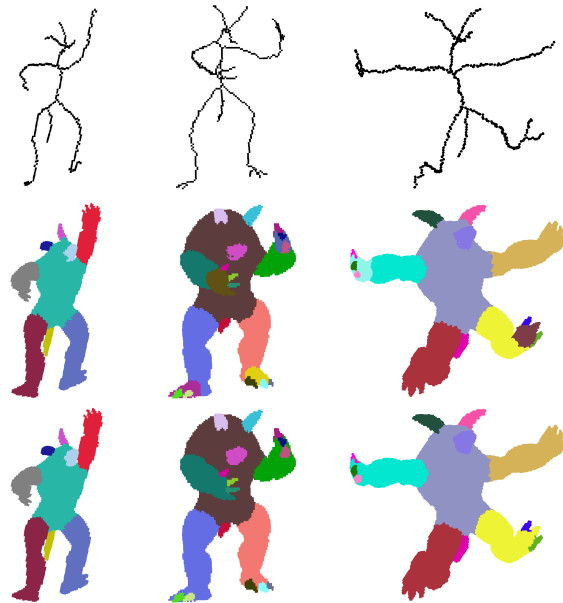


Fig. 5. From top to bottom: The skeletons of various input objects, the preliminary decompositions, and the resulting object decompositions

The algorithm runs on a Pentium 4 (3 GHz, 2 GB RAM) personal computer and its computational cost is  $O(N)$ , where  $N$  is the number of voxels in the image. The decomposition method is simple to implement, is rather fast and is completely automatic since it does not require any threshold. Of course, the quality of the results is influenced by the quality of the starting skeleton. In this respect, the skeletonization algorithm [17] has a positive impact on the decomposition method. In fact, due to the

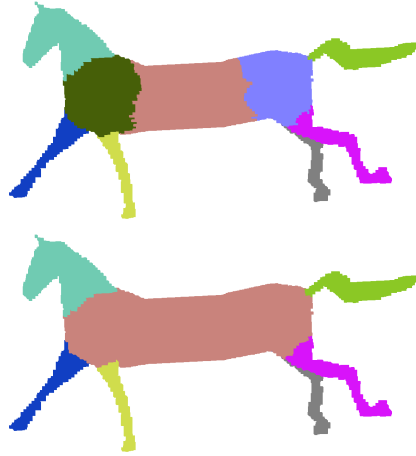
use of the  $\langle 3,4,5 \rangle$  distance that provides a good approximation of the Euclidean distance, the skeleton is rather stable under object rotation and for scale changes. Thus, stability also characterizes object decomposition. Moreover, the skeletonization algorithm includes a clever pruning step that, if the same object is presented in a different pose, allows us to obtain a skeleton with mostly the same structure. Actually, in this respect a key role for a satisfactory object decomposition is played by the zones of influence. In fact the skeleton of the same object in different poses may be characterized by a different number of branch points. However, due to the zones of influence we detect the same number of branch point configurations in all cases and, hence, the same number of perceptually significant skeleton branches.

Stability of the decomposition method with respect to pose/size changes can be appreciated by referring to Fig. 6, for the object “armadillo”. We observe that the main parts of the armadillo (the torso, the four limbs, the tail, the ears and the muzzle) are detected in all poses as individual decomposition parts. This is due to the detection of the zones of influence, which identifies the same number of branch point configurations in all cases, and to the criterion adopted to assign the overlapping regions to the proper adjacent components of object voxels. In turn, small peripheral parts, such as the toes, are not always individually detected as object parts since they are not individually mapped into skeleton branches.



**Fig. 6.** The skeleton of “armadillo” in different poses/sizes, top, the preliminary decomposition, middle, and the decomposition after assignment of the overlapping regions, bottom

We have also tested stability of decomposition when the object is deformed, e.g., by stretching it without tearing it apart or sticking distinct parts together. For example, see Fig. 7, where the decomposition of a deformed version of “horse” is shown. We note that also in this case two overlapping regions are found, which are still assigned to the torso.



**Fig. 7.** The preliminary decomposition of a deformed version of “horse”, top, and the decomposition into the same number of disjoint parts as that achieved for the non deformed “horse”, bottom

As pointed out in the introduction, this decomposition method has some analogies with our previous decomposition method [11]. Both methods decompose the skeleton and generate object decompositions into disjoint parts. Moreover, the overlapping regions and the components of object voxels obtained by subtraction of the overlapping regions from the input have some analogy with the kernels and with the simple regions and bumps detected in [11]. However, the two methods are rather different, both as concerns the model and as concerns the computational cost.

As concerns the model, in [11] we give a prominent role to the kernels, while in this paper the key role is played by the object parts that correspond to the perceptually meaningful skeleton branches. The one-to-one correspondence between the skeleton components identified during the skeleton decomposition process and the parts into which the object is decomposed is maintained by the current process, which produces as many object parts as many are the detected perceptually significant skeleton branches. In turn, with the method in [11] the one-to-one correspondence is maintained only if no merging phase is accomplished; on the other hand, merging is almost always necessary in order to obtain a decomposition more in accordance with intuition.

As regards the computational cost, the current method is noticeably cheaper. The object parts are simply obtained by subtraction from the input of the overlapping regions, while a 2-step more sophisticated and expensive process was used in [11] to build the various regions starting from the skeleton components. Moreover, a concavity filling algorithm had to be used in [11] to move a number of voxels from any kernel to the adjacent regions, so as to have almost planar separation cuts where significant changes of curvature occurred along the boundary of the object. In turn, this is no longer necessary in the new method, due to the criterion adopted to identify the overlapping regions. Finally, merging was a necessary step in [11], which implies

an additional computational effort and the use of merging thresholds, while such a step is not necessary in the current work.

As already said, our method effectively works for objects perceived as composed by the superposition of parts with tubular shape, possibly characterized by different width. According to our method, the overlapping regions cannot individually exist, nor are divided among the adjacent components of object voxels. The component to which an overlapping region is ascribed is the one whose shape appears as completed by the assignment of the overlapping region. Such a component is characterized by width comparable to the width of the adjacent overlapping region, while the remaining adjacent components have smaller width.

To extend the applicability of the method to an image domain wider than that including only objects that are perceived as composed by parts with tubular shape, some considerations on the assignment of the overlapping regions can be done. For example, think of a rounded pincushion from which a number of pins come out. We can assume that a unique large overlapping region exists, which exhausts the portion of the 3D space occupied by the cushion. If such an overlapping region is assigned to one of the pins, the shape of the so obtained object part would have no perceptual evidence. The boundary of the compound part (cushion plus pin), in fact, would not be characterized by that good continuity that a human observer would consider adequate to perceive that compound object part as a whole. A feature that would certainly characterize the adjacency matrix built for the pincushion is that the ratio  $R$  remains always rather small, since the cushion intrudes very little within the pins. Thus, a way to extend the decomposition method to a wider domain is to introduce a threshold on the minimal value that the ratio  $R$  should have in order a compound region (overlapping region plus adjacent component of object voxels) can be reasonably perceived as a whole. Selecting the proper value for such a threshold will be argument of future investigations. If for all components of object voxels adjacent to a given overlapping region the ratio  $R$  is below the threshold value, the overlapping region is not assigned to any component and is taken as an individual decomposition part. Obviously, the method would be no longer fully automatic, since a threshold has to be set, the one-to-one correspondence between perceptually meaningful skeleton branches and object parts would be no longer guaranteed, since overlapping regions may be individual decomposition parts, but the method would have larger applicability.

## 5 Concluding Remarks

In this work we have introduced a 3D object decomposition method based on skeleton decomposition. The objects of interest are understood as constituted by parts with tubular shape and possibly different width. Starting from the linear skeleton of the object, the zones of influence, i.e., the regions where different skeleton branches meet, are identified. The zones of influence are used to group branch points sufficiently close to each other, and to identify the perceptually meaningful skeleton branches. Then, the overlapping regions, i.e., the object regions where object parts

intersect, are identified and components of object voxels are obtained by subtraction of the overlapping regions from the input object. The overlapping regions are finally ascribed to the adjacent components of object voxels that better benefit of such an assignment.

## References

1. Palmer, S.E.: Hierarchical structure in perceptual representation. *Cognitive Psychology* 9, 441–474 (1977)
2. Marr, D., Nishihara, H.K.: Representation and recognition of three-dimensional shapes. *Proc. Royal Society of London: Series B* 200, 269–294 (1978)
3. Hoffman, D.D., Richards, W.A.: Parts of recognition. *Cognition* 18, 65–96 (1984)
4. Biederman, I.: Recognition-by-components: A theory of human image understanding. *Psychological Review* 94, 115–147 (1987)
5. Sanniti di Baja, G., Thiel, E.: The path-based distance skeleton: a flexible tool to analyse silhouette shape. In: *Proc. 12th ICPR*, vol. II, pp. 570–572 (1994)
6. Sanniti di Baja, G., Thiel, E.: (3,4)-weighted skeleton decomposition for pattern representation and description. *Pattern Recognition* 27, 1039–1049 (1994)
7. Cornea, N.D., Silver, D., Yuan, X., Balasubramanian, R.: Computing hierarchical curve-skeletons of 3D objects. *The Visual Computer* 21(11), 945–955 (2005)
8. Lien, J.-M., Geysler, J., Amato, N.M.: Simultaneous shape decomposition and skeletonization. In: *Proc. 2006 ACM Symposium on Solid and Physical Modeling*, pp. 219–228 (2006)
9. Reniers, D., Telea, A.: Skeleton-based hierarchical shape segmentation. In: *Proc. IEEE Int. Conf. on Shape Modeling and Applications*, pp. 179–188 (2007)
10. Serino, L., Sanniti di Baja, G., Arcelli, C.: Object decomposition via curvilinear skeleton partition. In: *Proc. 20th ICPR*, pp. 4081–4084 (2010)
11. Serino, L., Sanniti di Baja, G., Arcelli, C.: Using the Skeleton for 3D Object Decomposition. In: Heyden, A., Kahl, F. (eds.) *SCIA 2011*. LNCS, vol. 6688, pp. 447–456. Springer, Heidelberg (2011)
12. Chiang, P.-Y., Jay Kuo, C.-C.: Voxel-based shape decomposition for feature-preserving 3D thumbnail creation. *J. Vis. Commun. Image R.* 23, 1–11 (2012)
13. Borgefors, G.: On digital distance transform in three dimensions. *CVIU* 64(3), 368–376 (1996)
14. Blum, H.: Biological shape and visual science. *J. Theor. Biol.* 38, 205–287 (1973)
15. Nyström, I., Borgefors, G.: Synthesising Objects and Scenes Using the Reverse Distance Transformation in 2D and 3D. In: Braccini, C., Vernazza, G., DeFloriani, L. (eds.) *ICIAP 1995*. LNCS, vol. 974, pp. 441–446. Springer, Heidelberg (1995)
16. Svensson, S., Sanniti di Baja, G.: Using distance transforms to decompose 3D discrete objects. *Image and Vision Computing* 20, 529–540 (2002)
17. Arcelli, C., Sanniti di Baja, G., Serino, L.: Distance driven skeletonization in voxel images. *IEEE Trans. PAMI* 33(4), 709–720 (2011)
18. Shilane, P., Min, P., Kazhdan, M., Funkhouser, T.: The Princeton Shape Benchmark, Shape Modeling International, Genova, Italy (June 2004)

# Merging Faces: A New Orthogonal Simplification of Solid Models

Irving Cruz-Matías and Dolors Ayala

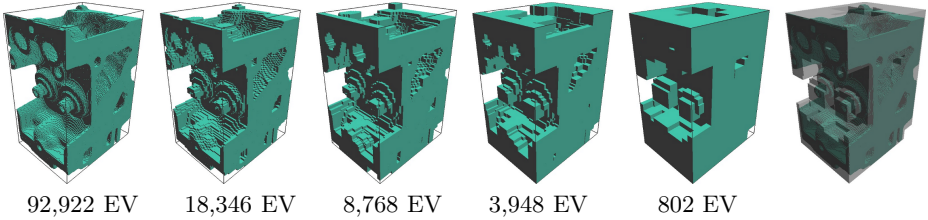
Department de Llenguatges i Sistemes Informàtics,  
Universitat Politècnica de Catalunya, Barcelona, Spain  
{icruz,dolorsa}@lsi.upc.edu

**Abstract.** A new approach to simplify orthogonal pseudo-polyhedra (OPP) and binary volumes is presented. The method is incremental and produces a level-of-detail (LOD) sequence of OPP. Any object of this sequence contains the previous objects and, therefore, it is a bounding orthogonal approximation of them. The sequence finishes with the minimum axis-aligned bounding box (AABB). OPP are represented by the Extreme Vertices Model, a complete model that stores a subset of their vertices and performs fast Boolean operations. Simplification is achieved using a new approach called *merging faces*, which relies on the application of 2D Boolean operations. We also present a technique, based on the model continuity, for a better shape preservation. The method has been tested with several datasets and compared with two similar methods.

**Keywords:** Simplification, LOD, Bounding Volumes, Orthogonal Polyhedra, Binary volumes.

## 1 Introduction

The large size and complexity of the models often affects the computation speed-up of their characteristics and their rendering efficiency. Simplification techniques can diminish these problems. Moreover, in some situations it is advantageous to exchange an exact geometric representation of an object for an approximated one, which can be processed more efficiently. Bounding structures are used for model simplification to accelerate tasks such as collision detection or distance computation. The most used bounding structures are AABB, spheres, oriented boxes or convex polyhedra. In this paper we present an approach to simplify OPP. The method computes a LOD sequence of bounding volumes (BV), that are also OPP, denoted as bounding OPP (BOPP). BOPP satisfy the two following properties: (1) any BOPP contains the previous one and (2) all the BOPP, as well as the original object, have the same AABB. Fig. 1 depicts a 3D model and the obtained BOPP sequence, with the AABB for each one. We use the Extreme Vertices Model (EVM) to represent OPP. The presented simplification approach, called *merging faces*, relies on the application of 2D Boolean operations, which are fast using EVM, over the OPP faces. The presented method deals with general 3D orthogonal objects with any number of shells, cavities and through



**Fig. 1.** A level-of-detail sequence of orthogonal pseudo-polyhedra (OPP) generated by our approach. From left to right: Original model, OPP with 19.7%, 9.4%, 4.2% and 0.8% of extreme vertices (EV). In the last one, the 1st and 5th objects are put together.

holes. We also develop a technique for a better shape preservation that avoids abrupt changes. The method has been tested with several datasets and compared with similar methods, showing satisfactory results.

## 2 Related Work

Model simplification has been extensively applied to triangular meshes [5] and extended to tetrahedral meshes evaluating the approximation error and the quality of the obtained mesh [4]. Methods for LOD sequences of triangular and tetrahedral meshes can also be found extensively in the literature [16] as well as methods to simplify quadrilateral meshes [11,21]. In contrast to these methods, that rely on geometric operations as edge-collapse or clustering, simplification can follow other strategies. Morphological operators as filleting and rounding can be used to simplify 2D binary images as well as 3D triangular meshes [26]. A carving strategy is applied to an octree model [23] as well as to a tetrahedral mesh [12] to simplify the topology. Simplification strategies have also been developed for B-Rep models [19] by removing connected sets of faces.

Several applications as collision detection [14], ray tracing [25] and volume of interest computation [9] use approximated shapes that are BV. Simple spheres [10] and AABB [20] are used as well as more sophisticated shapes as convex [15] or oriented [6] polytopes. Orthogonal polyhedra have also been proposed as BV [8] and as geometric bounds for CSG [1]. Orthogonally convex polygons are computed as orthogonal hulls for 2D images [3]. An orthogonal polygon is orthogonally convex if any axis-parallel line intersects it in at most one line segment. This problem has been extended to orthogonally convex polyhedra [2].

A sequence of BV can be obtained using alternative representations as octrees [18,23] resulting in a simplified geometry and topology, or BSP [13] obtaining a LOD sequence with a decreasing number of nodes. OP simplification has been carried out with a *moving faces* strategy that performs face displacements (but fails for objects with holes or more than one connected component) and the *rectangle pairs* strategy based on a box partition of the OP [8].



### 3 Extreme Vertices Model (EVM)

OP are two-manifold polyhedra with all their faces oriented in the three main axes. OPP are regular OP with a possible non-manifold boundary. General OPP are OPP with vertex coordinates having any value in  $\mathbb{R}^3$ . Polycubes are a subset of OPP all of whose vertices have integer coordinates, formed by joining one or more equal cubes (voxels) face to face. A 3D binary digital image represents an object as the union of its foreground voxels and its continuous analog is an OPP. In this paper we work with all these kind of objects represented with EVM. Let  $Q$  be a finite set of points in  $\mathbb{R}^3$ , the  $ABC$ -sorted set of  $Q$  is the set resulting from sorting  $Q$  according to  $A$ -coordinate, then to  $B$ -coordinate, and then to  $C$ -coordinate. Let  $P$  be an OPP, a *brink* is the maximal uninterrupted segment built out of a sequence of collinear and contiguous two-manifold edges of  $P$  and its ending vertices are called *extreme vertices* (EV). An OPP can be represented in a concise way with the  $ABC$ -sorted set of its EV and such representation scheme is called EVM. EVM is a complete solid model [24].

Let  $P$  be an OPP and  $\Pi_c$  a plane whose normal is parallel, without loss of generality, to the  $X$  axis, intersecting it at  $x = c$ , where  $c$  ranges from  $-\infty$  to  $\infty$ . Then, this plane sweeps the whole space as  $c$  varies within its range, intersecting  $P$  at some intervals. Let us assume that this intersection changes at  $c = c_1, \dots, c_n$ . More formally,  $P \cap \Pi_{c_i-\delta} \neq P \cap \Pi_{c_i+\delta}$ ,  $i = 1, \dots, n$ , where  $\delta$  is an arbitrarily small quantity. Then,  $C_i(P) = P \cap \Pi_{c_i}$  is called a *cut* of  $P$  and  $S_i(P) = P \cap \Pi_{c_s}$ ,  $c_i < c_s < c_{i+1}$ , is called a *section* of  $P$ . Two cuts bounding a section  $C_i$  and  $C_{i+1}$  are called *consecutive cuts*. See Fig. 2. Sections can be computed from cuts and vice versa:

$$\overline{S_0(P)} = \overline{S_n(P)} = \emptyset, \quad \overline{S_i(P)} = \overline{S_{i-1}(P)} \otimes^* \overline{C_i(P)}, i = 1 \dots n - 1 \quad (1)$$

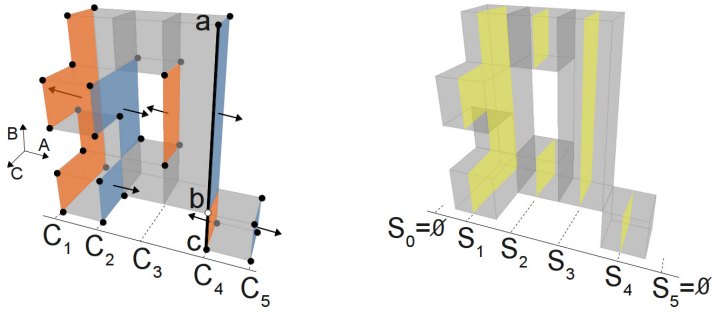
$$\overline{C_i(P)} = \overline{S_{i-1}(P)} \otimes^* \overline{S_i(P)}, i = 1 \dots n \quad (2)$$

where  $n$  is the number of cuts and  $\otimes$  denotes the xor operation. Overline symbolizes the project operator, that projects a  $d$ -dimensional set of vertices lying on an orthogonal plane, like a cut or a section, onto the corresponding main plain, discarding their  $d$ th coordinate. The star exponent  $*$  denotes a regularized Boolean operation. A regularized set is defined as the closure of its interior. Regularized Boolean operations are needed to ensure 3D homogeneity [22].

Eq. 2 can be rewritten by expressing the  $\otimes^*$  operation as the union of differences:  $\overline{C_i(P)} = (\overline{S_{i-1}(P)} -^* \overline{S_i(P)}) \cup^* (\overline{S_i(P)} -^* \overline{S_{i-1}(P)})$ , and any cut can be decomposed into its *forward difference* ( $FD$ ) and *backward difference* ( $BD$ ):

$$\overline{FD_i(P)} = \overline{S_{i-1}(P)} -^* \overline{S_i(P)}, \quad \overline{BD_i(P)} = \overline{S_i(P)} -^* \overline{S_{i-1}(P)}, i = 1 \dots n \quad (3)$$

$FD_i(P)$  is the set of  $C_i(P)$  faces whose normal vector points to the positive side of the coordinate axis perpendicular to  $C_i(P)$  and  $BD_i(P)$  is the set of faces whose normal vector points to the negative side (see Fig. 2). This property guarantees the correct orientation of faces and the computation of the non-EV. EVM Boolean operations are computed by applying recursively (in  $nD$ ) the same Boolean operation over the  $(n-1)D$  OPP sections. The base case performs this



**Fig. 2.** An EVM-encoded ( $ABC$ -sorted) OPP. Left: EV marked with dots, cuts decomposed in FD (blue) and BD (orange), normal vectors represented with arrows; a vertical brink from vertex  $a$  to  $c$  is marked showing that these vertices are both EV while the vertex  $b$  is a non-EV. Right: The sections of the object highlighted in yellow.

operation in 1D. The  $\otimes^*$  is even faster,  $EVM(P \otimes^* Q) = EVM(P) \otimes^* EVM(Q)$ , i.e, it is a simple point-wise xor, without section computation. For more details concerning EVM see [1] and [17].

### 4 Algorithm Overview

Let  $B(P)$  be the bounding orthogonal pseudo-polyhedron (BOPP) of an OPP  $P$  and let  $\phi_0$  be an initial OPP. A finite sequence  $\phi_1, \phi_2, \dots, \phi_p$  of OPP is generated that fulfill the following properties:

1.  $\phi_{i+1} = B(\phi_i), i = 0 \dots p - 1$
2.  $\phi_i \subseteq B(\phi_i), i = 0 \dots p - 1$ , and, therefore,  $\phi_i \subseteq \phi_{i+1}, i = 0 \dots p - 1$
3.  $\phi_p = AAB B(\phi_i), i = 0 \dots p$

The first property indicates that the approach is incremental. The second one, called subset property, is intrinsic in bounding structures. The last property states that the sequence is finite and that ends with the AAB B that is shared by all the OPP of the sequence.

The simplification strategy, *merging faces*, works with pairs of consecutive cuts of  $P$ . For each cut of a pair, a displacement to its faces in the direction of their corresponding normal vector is applied and, then, the displaced faces are merged with those faces of the other cut with the same normal vector. The process is controlled by the displacement parameter,  $d$ , that indicates the maximum displacement allowed, in such a way that only pairs of consecutive cuts that are at a distance  $\leq d$  are actually merged. If  $P$  is represented as an  $ABC$ -sorted EVM, the application of this process only will coarsen  $P$  in the  $A$ -coordinate. Therefore to obtain  $B(P)$  the process is repeated for the other two main directions. The result can be slightly different depending on the ordering in which the three main directions are selected. In order to speed up the computation, the best first candidate would be the  $ABC$ -ordering with  $A$ -axis having less number of

cuts, but as EVM does not report this value, it can be approximated by  $c_n - c_1$ ,  $c_1$  and  $c_n$  being respectively the coordinates of the first and last cut in this direction. We can compute the whole sequence of BOPP or ask for a BOPP with a maximum number of EV.

The input of the whole method is an EVM represented OPP,  $P$ , which corresponds to the initial object  $\phi_0$  and the desired maximum number of EV,  $nv$ , in the simplified model, and returns an EVM represented OPP,  $Q$ , corresponding to the object  $\phi_k$  of the sequence, which has no more than  $nv$  EV. As the method is incremental, it actually computes all the objects between  $\phi_1$  and  $\phi_k$ . It performs  $k$  times the *merging faces* process for successive values of  $d$ ,  $d = d_i, i = 1, \dots, k$ , obtaining the corresponding  $\phi_i$ . Displacements  $d_i$  can be in any units and incremented in any quantity. For OPP corresponding to digital images the basic unit is one voxel, i.e.  $d_i = i$ . For general OPP, with float coordinate values,  $d_i$  can take any values ranging from the minimum distance between cuts and the AABB size. Note that if the increment of  $d$  is too large, far apart consecutive cuts can be merged in early iterations causing abrupt changes in the simplified object. The next pseudocode shows the iterative algorithm and concerning  $d$  considers digital images. *Get\_nev()* returns the number of EV of the given object  $P$ , *mergingFaces()* receives the object  $P$  and a distance  $d$ , and returns the object  $Q = B(P)$ , merging pairs of consecutive cuts at a distance  $\leq d$ . To compute the whole LOD sequence of BOPP,  $nv$  must be 8.

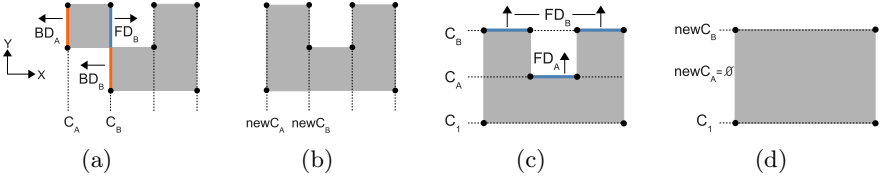
```
EVM Simplification(EVM P, int nv) {
    EVM Q = P;
    for(int d=1; Q->Get_nev() > nv; d++){
        Q->SetSorting(ABC);    Q = mergingFaces(Q, d);
        Q->SetSorting(BAC);    Q = mergingFaces(Q, d);
        Q->SetSorting(CAB);    Q = mergingFaces(Q, d); }
    return Q; }
```

## 5 Merging Faces Approach

In this section we first explain the basic process, then the treatment of the void space and finally, we discuss the way to select pairs of cuts. Let  $P$  be an OPP and let  $C_A$  and  $C_B$  be two consecutive cuts of  $P$  with  $FD_A, BD_A, FD_B, BD_B$  as their corresponding forward and backward differences (see Eq. 3). To obtain a coarsened OPP, the *merging faces* process displaces  $BD_B$  to the position of  $BD_A$ , and  $FD_A$  to the position of  $FD_B$ . Then the new cuts  $newC_A$  and  $newC_B$  that will replace  $C_A$  and  $C_B$ , respectively, in the input model, are computed as:

$$\overline{newC_A} = \overline{BD_A} \cup^* \overline{BD_B}, \quad \overline{newC_B} = \overline{FD_A} \cup^* \overline{FD_B} \quad (4)$$

This process fulfills the properties stated in Sec. 4. Face displacements are done in the direction of their respective normal vector, i.e. outward of the object. Then, for any  $P$ ,  $P \subseteq B(P)$  (subset property). The property concerning AABB can be proved by considering that the AABB of an OPP can be defined as



**Fig. 3.** 2D example: (a) Faces merged in a first step. (b) Result after applying Eq. 4 and end for XY-sorting. (c) Faces merged by YX-sorting. (d) Resulting BOPP.

the intersection of the six planes corresponding to the first and last cut in the three main directions, and the fact that the object will never extend beyond the first and last cut of the initial object because the displacements are bounded by these cuts. Moreover, in the last iteration, the first cut of  $\phi_p$  (the AABB), will correspond to the union of all BD of all cuts, and likewise the last cut to the union of all FD. Fig. 3 shows a 2D example where the process is applied.

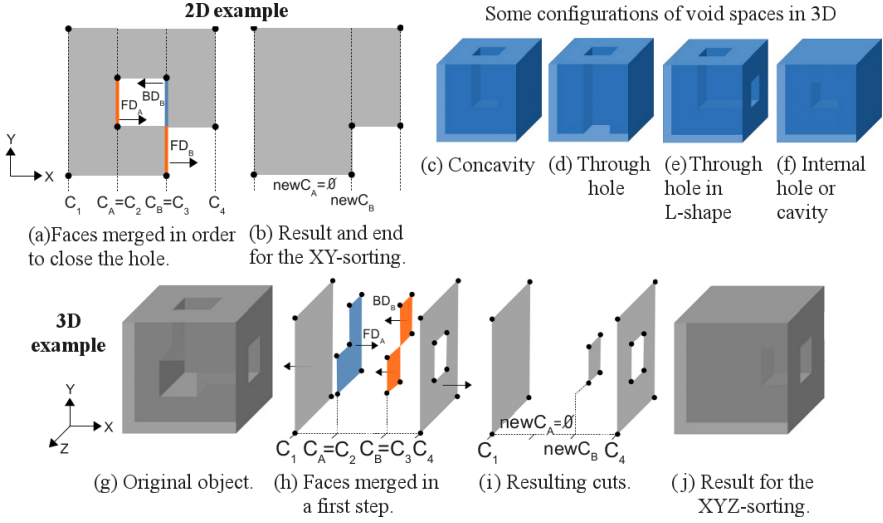
An OPP may have any number of rings on faces, through holes and shells. In these cases, Eq. 4 could not give the desirable result, for instance, the object in Fig. 4(a), remains the same after the application of Eq. 4. To deal with this issue, we first detect and then close void spaces. Given  $C_A$  and  $C_B$ ,  $BD_A$  and  $BD_B$  are sets of faces whose normal vectors point to the opposite direction of  $FD_A$  and  $FD_B$ , but  $FD_A$  and  $BD_B$  define a void space as  $vSpace(C_A, C_B) = \overline{FD_A} \cap^* \overline{BD_B}$ . Removing  $vSpace$  from both  $newC_A$  and  $newC_B$ , closes the void space between  $C_A$  and  $C_B$ . Then, Eq. 4 is extended as:

$$\begin{aligned} \overline{newC_A} &= (\overline{BD_A} \cup^* \overline{BD_B}) -^* (\overline{FD_A} \cap^* \overline{BD_B}) \\ \overline{newC_B} &= (\overline{FD_A} \cup^* \overline{FD_B}) -^* (\overline{FD_A} \cap^* \overline{BD_B}) \end{aligned} \quad (5)$$

Observe that when  $vSpace = \emptyset$ , Eq. 5 and 4 are equivalent. The subset and the AABB properties are also guaranteed as we remove interior void spaces. Fig. 4(a) shows an example with a single hole, here  $\overline{FD_A} = \overline{BD_B}$  and  $vSpace(C_A, C_B) = \overline{FD_A} = \overline{BD_B}$ . Then, applying Eq. 5,  $newC_A = \emptyset$  and the hole is closed (Fig. 4(b)). In Eq. 5, basic merging and void space removal are performed jointly. Some void spaces, as the simple concavity in Fig. 4(c), are solved with Eq. 4. However, general void spaces, as those in Fig. 4(d-f), require the application of Eq. 5. Some void spaces can be detected in all three directions, as the cavity depicted in Fig. 4(f) but some others are only detected in one or two directions. Fig. 4(g-j) show a working example in an object with through holes.

Two EVM properties, that state that for two special cases union and difference consist in simple point-wise xor operations [1], together with two EVM theorems, permit to rewrite Eq. 5 in a way faster to compute.

- *Property 1:* Let  $P$  and  $Q$  be two OPP such that  $P \cap^* Q = \emptyset$ , having  $EVM(P)$  and  $EVM(Q)$  as their models, then  $EVM(P \cup^* Q) = EVM(P) \otimes^* EVM(Q)$ .
- *Property 2:* Let  $P$  and  $Q$  be two OPP such that  $P \supseteq Q$ , with  $EVM(P)$  and  $EVM(Q)$  as their models, then  $EVM(P -^* Q) = EVM(P) \otimes^* EVM(Q)$ .



**Fig. 4.** *Merging faces* and treatment of void space

**Theorem 1.** *The projection of  $FD$  and  $BD$  of two consecutive cuts  $C_A$  and  $C_B$  are quasi-disjoint sets respectively, i.e.  $\overline{FD_A} \cap^* \overline{FD_B} = \emptyset$  and  $\overline{BD_A} \cap^* \overline{BD_B} = \emptyset$ .*

*Proof.* The proof is based on the Jordan theorem and the fact that any ray crossing the boundary of the polyhedron, alternatively goes from outside to inside and vice versa. Therefore, in any OPP, assuming that  $\overline{FD_A} \cap^* \overline{FD_B} \neq \emptyset$ , would mean that a ray could cross  $FD_A$  going outside and then cross  $FD_B$  going outside again, which is a contradiction. The same reasoning applies to  $BD$ .  $\square$

**Theorem 2.**  $vSpace(C_A, C_B) \subseteq (\overline{BD_A} \cup^* \overline{BD_B})$

*Proof.*  $vSpace(C_A, C_B) = \overline{FD_A} \cap^* \overline{BD_B}$ . Without loss of generality:  $(\overline{FD_A} \cap^* \overline{BD_B}) \subseteq \overline{BD_B} \subseteq (\overline{BD_A} \cup^* \overline{BD_B})$ , and thus:  $(\overline{FD_A} \cap^* \overline{BD_B}) \subseteq (\overline{BD_A} \cup^* \overline{BD_B})$

In a similar way  $vSpace(C_A, C_B) \subseteq (\overline{FD_A} \cup^* \overline{FD_B})$  can be proved.  $\square$

According to these theorems and the EVM properties, Eq. 5 is rewritten as:

$$\begin{aligned} \overline{newC_A} &= \overline{BD_A} \otimes^* \overline{BD_B} \otimes^* (\overline{FD_A} \cap^* \overline{BD_B}) \\ \overline{newC_B} &= \overline{FD_A} \otimes^* \overline{FD_B} \otimes^* (\overline{FD_A} \cap^* \overline{BD_B}) \end{aligned} \quad (6)$$

*Merging faces* takes pairs of consecutive cuts but we must establish the way in which they are selected. After analyzing several alternatives, we selected the one with best visual results. It consists in taking cuts two by two:  $(C_A = C_i, C_B = C_{i+1})$ ,  $i = 1, i \leq n - 1, i = i + 2$ , i.e. first the pair  $(C_1, C_2)$ , then  $(C_3, C_4)$ , and so on. However, when  $C_A$  and  $C_B$  are such that  $FD_A = BD_B = \emptyset$ , *merging faces* has no effect (e.g. consider  $C_A = C_1$  and  $C_B = C_2$  in Fig. 4(h)). On the other hand,

only pairs of cuts at a distance  $\leq d$  are processed. Then, given the pair of cuts ( $C_A=C_i, C_B=C_{i+1}$ ), if Exp. 7 is fulfilled, this pair is processed and the method continues with the pair ( $C_{i+2}, C_{i+3}$ ). Otherwise  $C_i$  is copied to the resulting OPP and the method continues with the pair ( $C_{i+1}, C_{i+2}$ ). Moreover, for symmetry preservation purposes, the model is analyzed from both sides at a time.

$$(FD_A \neq \emptyset \text{ or } BD_B \neq \emptyset) \text{ and } distance(C_A, C_B) \leq d \quad (7)$$

## 6 Shape Preservation

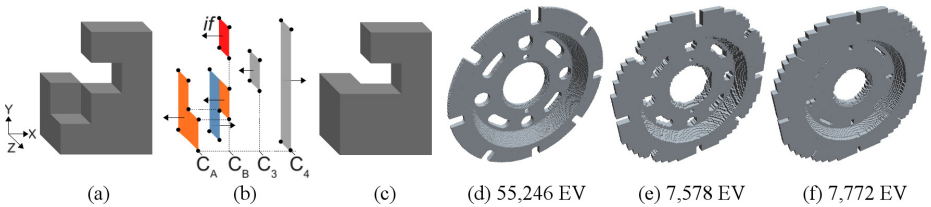
Shape preservation is an important aspect in model simplification. Merging two cuts is performed in all their extension, but there are parts of these cuts that are isolated and merging them could result in too abrupt changes. Let ( $C_A, C_B$ ) be the pair of cuts to be merged, we will refer as *isolated faces* those faces of  $C_A$  whose projections do not share either an edge or a vertex with others in  $C_B$  and vice versa. Removing these faces of the merging process results in a better approximation (see Fig. 5(d-f)). Let  $I_A$  and  $I_B$  be the *isolated faces* of  $C_A$  and  $C_B$  respectively, the new  $FD'_A, BD'_A, FD'_B$  and  $BD'_B$  are computed according to Eq. 8. These values are used in Eq. 6 to generate  $newC'_A$  and  $newC'_B$ , and after that, the isolated faces can be reintegrated with a union operation. However, by definition  $I_A$  and  $C_A$  are disjoint sets and therefore,  $I_A$  and  $newC'_A$  are also disjoint sets (the same applies to  $I_B, C_B$  and  $newC'_B$ ). Then, according to *Property 1*, an xor operation is performed instead (See Eq. 9).

$$\overline{FD'_A} = \overline{FD_A} - * \overline{I_A}, \quad \overline{BD'_A} = \overline{BD_A} - * \overline{I_A} \quad (8)$$

$$\overline{FD'_B} = \overline{FD_B} - * \overline{I_B}, \quad \overline{BD'_B} = \overline{BD_B} - * \overline{I_B}$$

$$\overline{newC'_A} = \overline{newC'_A} \otimes * \overline{I_A}, \quad \overline{newC'_B} = \overline{newC'_B} \otimes * \overline{I_B} \quad (9)$$

Fig. 5(a-c) show an example where an isolated face (*if*) is depicted. Note that *if* remains in place, and only those faces that share either an edge or a vertex are taken into account throughout the *merging faces* process. The application of this technique does not affect the subset property (see Sec. 4). However, the property concerning finiteness cannot be guaranteed, as there may be consecutive



**Fig. 5.** *Shape preservation* technique. (a) A simple 3D model. (b) All cuts and EV; in red an isolated face. (c) Result of applying *merging faces* with the technique to the pair ( $C_A, C_B$ ). (d) Original DiskBrake model. (e) and (f) BOPP with and without the shape preservation technique respectively, both having less than 15% of EV.

cuts where all the faces are isolated and, in this case, we would not reach the AABB. Therefore, when all the faces are isolated, we do not apply the shape preservation technique and use directly Eq. 6. To detect isolated faces between  $C_A$  and  $C_B$ , we check each face of  $C_A$  against each face of  $C_B$ . For this process we use a decomposition model, the Compact Union of Disjoint Boxes (CUDB) [7], which is derived from EVM by splitting the orthogonal faces into a set of AB-sorted rectangles. We have implemented an algorithm for rectangle adjacency detection. As a 2D CUDB is a sorted set of rectangles, we apply a merging process, i.e., we perform a traversal of the CUDB models of each face in  $C_A$  and  $C_B$  at a time and check for rectangle intersection. Thus, if we have  $n$  faces with  $m_i$  boxes each one, the worst case time-complexity is  $O(n \cdot M)$ ,  $M = \sum_{i=1}^n m_i$ .

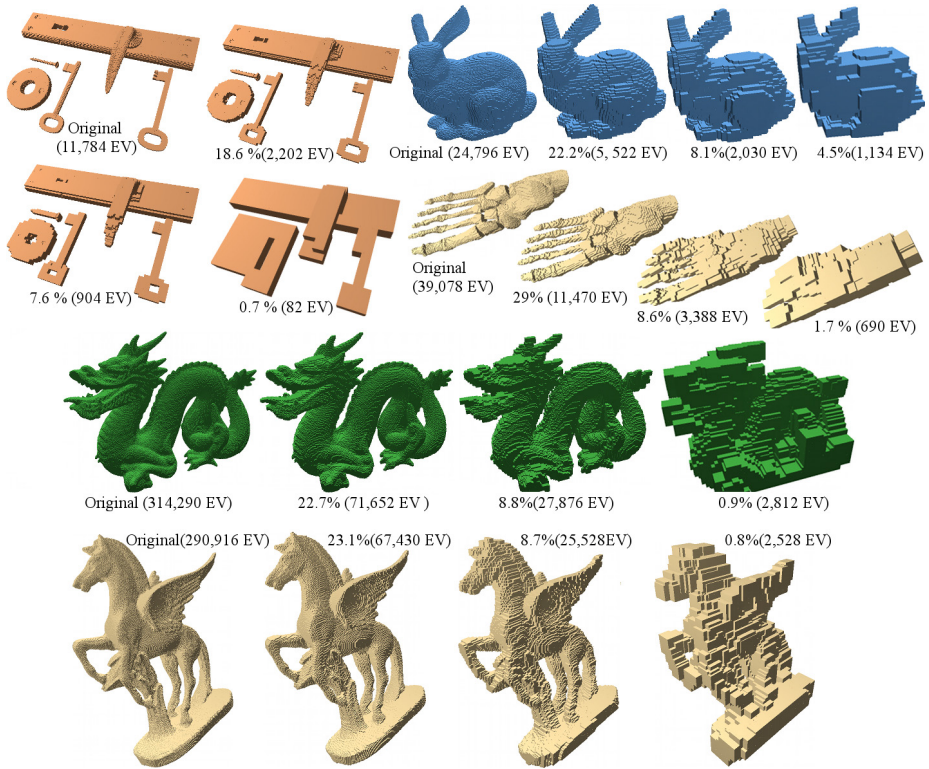
## 7 Results and Discussion

We have tested *merging faces* with and without shape preservation in several 3D datasets with different shape features, obtained from public repositories and our own collection. All of them have been converted to EVM using existing algorithms [17]. The presented algorithms have been written in C++ and executed on a PC Intel®Core i7 CPU 870 at 2.93GHz with 16Gb of RAM under Linux. The computation time of *merging faces* depends directly on the number of EV which is related to how well aligned is the object with respect to the three axes. This fact is shared by most of the EVM-based developed methods.

Table 1 shows the obtained results. Note that *merging faces* with shape preservation requires, in general, more time than without it. However, in some cases the time is almost the same (DiskBrake) or even less (Pegasus, Dragon). This is due to that in some datasets the application of shape preservation results in a significant reduction of the number of processed pairs and, consequently, in the execution time. Figures 1 and 6 depict some of these datasets with several BOPP of their LOD sequence using shape preservation. We consider EV as the basic geometric element and these figures show the number of EV and the percentage of reduction with respect to the number of EV the original dataset. We can observe that BOPPs with less than 25% of EV give a very good approximation of

**Table 1.** For each dataset: size in voxels and number of EV ( $|EV|$ ); maximum distance  $d$ , number of processed pairs  $np$  and time  $T$  to compute the whole LOD sequence without shape preservation;  $d_{sp}$ ,  $np_{sp}$  and  $T_{sp}$ : same values with shape preservation

Dataset	Size	$ EV $	$d$	$np$	$T$ (sec.)	$d_{sp}$	$np_{sp}$	$T_{sp}$ (sec.)
DoorPieces	267x394x72	11,784	46	894	<b>0.35</b>	46	953	0.47
Bunny	127x128x98	24,796	24	1,098	<b>0.63</b>	30	1,236	0.85
Foot	96x270x97	39,078	55	2,031	<b>1.32</b>	49	2,009	1.73
DiskBrake	299x300x43	55,246	32	2,335	<b>1.85</b>	35	2,173	1.95
Engine	140x197x108	92,922	19	2,189	<b>3.08</b>	22	2,761	4.95
Pegasus	382x512x367	290,916	103	20,782	37.85	137	9,228	<b>19.55</b>
Dragon	511x360x228	314,290	56	10,646	19.01	57	8,325	<b>18.42</b>

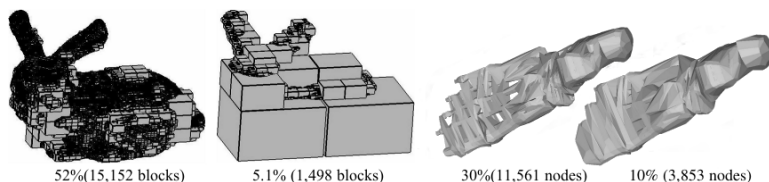


**Fig. 6.** Some results of the tested models, showing the number of extreme vertices

the objects, that BOPPs with less than 10% yield an acceptable approximation, and, although BOPPs with less than 1% look blocky, they still give an indication of the object shape. Also note how the holes progressively get closed, specially in Engine and DoorPieces and the connected components are joined in DoorPieces.

We have compared the presented method against two similar methods that compute bounding volumes (see Figs. 6 and 7). We compare the experimental results in terms of percentage of basic geometric elements reduction versus visual approximation. The first method (OCT) [18] is an octree-based approximation and compression method for 3D objects. Approximations are obtained by truncating the octree. The results of the method are presented in terms of number of blocks (octree nodes), which is the basic geometric element of the octree and the test model used is the Bunny dataset. The original model requires 29,007 blocks and 24,796 EV. Figures 7(left) show two objects generated by OCT. We can see that *merging faces* gives a better indication of the shape with less than 5.1% of elements than OCT, and even an object with 22.2% gives a better approximation than OCT with 52%. The second method (BSP)[13] is a progressive solid simplification of objects represented by a Binary Space Partition tree. It uses a volume bounded convex simplification and a plane collapse method to reduce





**Fig. 7.** Some results of OCT (Bunny dataset) and BSP (Foot dataset) methods, showing the number of elements, blocks and BSP nodes respectively. Figures have been taken from the references [18,13].

the BSP-tree depth. The basic geometric element is a plane (BSP node). We have used the most of the datasets presented in this work and converted them to EVM in such a way that the number of EV of the original model is approximately the same than the number of planes in the BSP tree. We report here the results of the Foot dataset. The original model has 39,078 EV and 38,535 planes. Figures 7(right) show two LOD-objects generated by BSP. We can observe that the shape of the fingers in *merging faces* is preserved until the 1.7% version while in the BSP method the fingers look wrapped since the 30% version. Similar results have been obtained with the other datasets.

## 8 Conclusions

We have proposed an approach to simplify orthogonal polyhedra and binary images, represented with the EVM. It generates a LOD sequence of BOPP, fulfilling the common subset and AABB properties of bounding structures. The approach is based on a merging strategy that involves pairs of consecutive cuts. We have showed that our approach can deal with objects with any number of holes and connected components, and presented a technique, based on the model continuity, for a better shape preservation. We have compared our method against two similar methods, and in general, our method gives better approximations with the same number of basic geometric elements. Directions for future work include the study of a lossless simplification approach based on the presented one.

**Acknowledgements.** This work was partially supported by the national projects TIN2008-02903 and TIN2011-24220 and a MAEC-AECID grant for I. Cruz-Matías, all of the Spanish government. The authors thank the anonymous reviewers whose remarks and suggestions have allowed to greatly improve the paper.

## References

1. Aguilera, A., Ayala, D.: Orthogonal Polyhedra as Geometric Bounds in CSG. In: IVth Symp. on Solid Modeling and Appl., pp. 56–67. ACM (1997)
2. Biedl, T., Geng, B.: Reconstructing orthogonal polyhedra from putative vertex sets. *Computational Geometry* 44(8), 409–417 (2011)

3. Biswas, A., et al.: A linear-time combinatorial algorithm to find the orthogonal hull of an object on the digital plane. *Information Sciences* 216, 176–195 (2012)
4. Cignoni, P., et al.: Simplification of tetrahedral meshes with accurate error evaluation. In: *Proceedings of the IEEE Visualization Conference*, pp. 85–92 (2000)
5. Cignoni, P., Montani, C., Scopigno, R.: A comparison of mesh simplification algorithms. *Computers and Graphics* 22(1), 37–54 (1998)
6. Coming, D.S., Staadt, O.G.: Velocity-aligned discrete oriented polytopes for dynamic collision detection. *IEEE Trans. Vis. and Computer Graphics* 14, 1–12 (2008)
7. Cruz-Matías, I., Ayala, D.: CUDB: An improved decomposition model for orthogonal pseudo-polyhedra. *Tech. Rep. LSI-11-2-T, UPC* (2011)
8. Esperança, C., Samet, H.: Orthogonal Polygons as Bounding Structures in Filter-Refine Query Processing Strategies. In: Scholl, M.O., Voisard, A. (eds.) *SSD 1997*. LNCS, vol. 1262, pp. 197–220. Springer, Heidelberg (1997)
9. Fuchs, R., Welker, V., Hornegger, J.: Non-convex polyhedral volume of interest selection. *J. of Computerized Medical Imaging and Graphics* 34(2), 105–113 (2010)
10. Gagvani, N., Silver, D.: Shape-based volumetric collision detection. In: *VVS 2000: Proceedings of the 2000 IEEE Symp. on Volume Visualization*, pp. 57–61. ACM Press (2000)
11. Groß, A., Klein, R.: Efficient representation and extraction of 2-manifold isosurfaces using kd-trees. *Graphical Models* 66, 370–397 (2004)
12. Hagbi, N., El-Sana, J.: Carving for topology simplification of polygonal meshes. *Comput. Aided Des.* 42, 67–75 (2010)
13. Huang, P., Wang, C.: Volume and complexity bounded simplification of solid model represented by binary space partition. In: *Proc. SPM 2010*, pp. 177–182 (2010)
14. Jiménez, P., Thomas, F., Torras, C.: 3D collision detection: A survey. *Computers and Graphics* 25(2), 269–285 (2000)
15. Klosowski, J.T., et al.: Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE Trans. Vis. and Computer Graphics* 4(1), 21–36 (1998)
16. Ripolles, O., Chover, M., Gumbau, J., Ramos, F., Puig, A.: Rendering continuous level-of-detail meshes by masking strips. *Graphical Models* 71(5), 184–195 (2009)
17. Rodríguez, J., Ayala, D., Aguilera, A.: EVM: A Complete Solid Model for Surface Rendering. In: *Geometric Modeling for Scientific Vis.*, pp. 259–274. Springer (2004)
18. Samet, H., Kochut, A.: Octree approximation and compression methods. In: *Proceedings of the 1st Int. Symp. 3DPVT*, pp. 460–469. IEEE Computer Society (2002)
19. Sun, R., Gao, S., Zhao, W.: An approach to b-rep model simplification based on region suppression. *Computers & Graphics* 34(5), 556–564 (2010)
20. Suri, S., Hubbard, P.M., Hughes, J.F.: Analyzing bounding boxes for object intersection. *ACM Trans. Graph.* 18, 257–277 (1999)
21. Tarini, M., Pietroni, N., Cignoni, P., Panozzo, D., Puppo, E.: Practical quad mesh simplification. *Computer Graphics Forum* 29(2), 407–418 (2010)
22. Tilove, R., Requicha, A.: Closure of boolean operations on geometric entities. *Computer-Aided Design* 12(5), 219–220 (1980)
23. Vanderhyde, J., Szymczak, A.: Topological simplification of isosurfaces in volume data using octrees. *Graphical Models* 70, 16–31 (2008)
24. Vigo, M., Pla, N., Ayala, D., Martínez, J.: Efficient algorithms for boundary extraction of 2D and 3D orthogonal pseudomanifolds. *Grap. Models* 74, 61–74 (2012)
25. Wald, I., Boulos, S., Shirley, P.: Ray tracing deformable scenes using dynamic bounding volume hierarchies. *ACM Trans. Graph.* 26 (January 2007)
26. Williams, J., Rossignac, J.: Tightening: Morphological simplification. *International Journal of Computational Geometry & Applications* 17(5), 487–503 (2007)

# Sufficient Conditions for Topological Invariance of 2D Images under Rigid Transformations<sup>\*</sup>

Phuc Ngo<sup>1</sup>, Yukiko Kenmochi<sup>1</sup>, Nicolas Passat<sup>2</sup>, and Hugues Talbot<sup>1</sup>

<sup>1</sup> Université Paris-Est, LIGM, UPEMLV-ESIEE-CNRS, France

<sup>2</sup> Université de Reims, CReSTIC, EA 3804, France

**Abstract.** In  $\mathbb{R}^2$ , rigid transformations are topology-preserving operations. However, this property is generally no longer true when considering digital images instead of continuous ones, due to digitization effects. In this article, we investigate this issue by studying discrete rigid transformations (DRTs) on  $\mathbb{Z}^2$ . More precisely, we define conditions under which digital images preserve their topological properties under any arbitrary DRTs. Based on the recently introduced notion of DRT graph and the classical notion of simple point, we first identify a family of local patterns that authorize topological invariance under DRTs. These patterns are then involved in a local analysis process that guarantees topological invariance of whole digital images in linear time.

**Keywords:** 2D digital image, discrete rigid transformation, topology, simple point, DRT graph, Eulerian model.

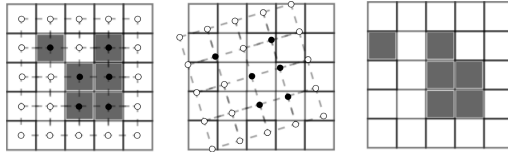
## 1 Introduction

In 2D, rigid transformations (*i.e.*, rotations composed with translations) are involved in numerous image processing/analysis tasks, *e.g.*, registration [1] or tracking [2]. In such applications, the images are generally digital, and can then be considered as functions  $I : S \rightarrow F$  from a finite subset  $S \subset \mathbb{Z}^2$  to a value space  $F$ . While rigid transformations are topology-preserving operations in  $\mathbb{R}^2$ , this property is generally lost in  $\mathbb{Z}^2$ , due to the discontinuities induced by the mandatory digitization from  $\mathbb{R}$  to  $\mathbb{Z}$ . In particular, discrete rigid transformations (DRTs) –that include discrete rotations [3,4,5,6]– are not guaranteed to preserve the homotopy type of digital images, as exemplified in Fig. 1.

In this article, we study this specific issue. More precisely, we investigate some conditions under which digital images preserve their topological properties under *any* arbitrary DRTs, by considering the Eulerian (*i.e.*, backwards) transformation model. To reach this goal, we consider (*i*) the notion of DRT graph, recently introduced by the authors in [7,8], that defines a combinatorial model of all the rigid transformations of a digital image, and (*ii*) the classical notion of simple point [9,10], that provides sufficient conditions to guarantee the preservation of homotopy type.

---

<sup>\*</sup> The research leading to these results has received funding from the French *Agence Nationale de la Recherche* (Grant Agreement ANR-2010-BLAN-0205 03).



**Fig. 1.** Left: a binary digital image and the grid modeling its discrete structure. Middle: a rigid transformation applied on this grid. Right: the resulting transformed image, with a homotopy different from the initial one (the black pixels, in the 8-adjacency, have been split).

By combining these two notions, we first propose a way to determine transformed images which have the same homotopy type as the initial one, by scanning the whole DRT graph associated to this image. Then, we show that this global approach, which presents a polynomial complexity, can be simplified into a local approach, based on a spatial decomposition of the image into covering samples. In order to do so, we identify a family of local patterns that authorize topological invariance under DRTs. These patterns can then be involved in a procedure based on look-up tables (LUT) that guarantee topological invariance of a whole digital image in linear time.

The article is organised as follows. Sec. 2 presents background notions related to rigid transformations and digital topology. Sec. 3 describes the topological issues induced by DRTs. Sec. 4 explains how DRT graphs and simple points can be combined to evaluate topological invariance under DRTs, leading to an algorithm detailed in Sec. 5. Experiments are proposed in Sec. 6, while Sec. 7 concludes the article.

## 2 Background Notions

### 2.1 (Discrete) Rigid Transformations

In  $\mathbb{R}^2$ , a rigid transformation (*i.e.*, a transformation composed of a translation and a rotation) is expressed as a bijection  $\mathcal{T} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$  defined, for any  $\mathbf{x} = (x, y) \in \mathbb{R}^2$  by

$$\mathcal{T}(\mathbf{x}) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} \quad \text{with } a, b \in \mathbb{R} \text{ and } \theta \in [0, 2\pi[ \quad (1)$$

Such a transformation (also noted  $\mathcal{T}_{ab\theta}$ ) is unambiguously modeled by the triplet of parameters  $(a, b, \theta)$ . It is not possible to apply directly  $\mathcal{T}$  on a digital image  $I : S \rightarrow F$ , since there is no guarantee that  $\mathcal{T}(\mathbf{x}) \in \mathbb{Z}^2$ , for any  $\mathbf{x} \in S \subset \mathbb{Z}^2$ . The handling of *discrete* rigid transformations (DRTs) then requires the definition of a function  $T : \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$ , which is the “discrete analogue” of  $\mathcal{T}$ . Considering the standard rounding function  $D : \mathbb{R}^2 \rightarrow \mathbb{Z}^2$ , this can be conveniently performed by setting  $T = D \circ \mathcal{T}$ , as illustrated on the diagram below.

$$\begin{array}{ccc}
\mathbb{Z}^2 & \xrightarrow{T=D \circ \mathcal{T}} & \mathbb{Z}^2 \\
\downarrow Id & & \uparrow D \\
\mathbb{R}^2 & \xrightarrow{\mathcal{T}} & \mathbb{R}^2
\end{array}$$

The function  $T : \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$  is then explicitly defined, for any  $\mathbf{p} = (p, q) \in \mathbb{Z}^2$ , by

$$T(\mathbf{p}) = D \circ \mathcal{T}(\mathbf{p}) = \begin{pmatrix} [p \cos \theta - q \sin \theta + a] \\ [p \sin \theta + q \cos \theta + b] \end{pmatrix} \quad (2)$$

In general, this function is not bijective. However, by setting  $T^{-1} = D \circ \mathcal{T}^{-1} : \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$ , we can define the transformed digital image  $I \circ T^{-1} : \mathbb{Z}^2 \rightarrow F$  with respect to  $T$ . Note that  $T^{-1}$  is not the inverse function of  $T$  in general.

## 2.2 Digital Topology

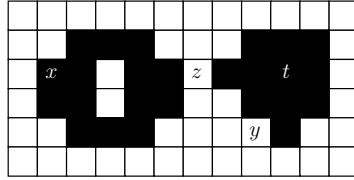
Several frameworks are available to model the topological structure of a digital image. In  $\mathbb{Z}^2$ , most of these frameworks (see, *e.g.*, [11,12]) can be conveniently unified within the frequently used –and also simple– framework of digital topology [9]. In this framework, the topological notions derive from a graph structure induced by two adjacency relations, namely the 4- and 8-adjacencies, which are defined for any two points  $\mathbf{p}, \mathbf{q} \in \mathbb{Z}^2$  such that  $\mathbf{p}$  and  $\mathbf{q}$  are 4-*adjacent* (resp. 8-*adjacent*) if  $\|\mathbf{p} - \mathbf{q}\|_1 \leq 1$  (resp. if  $\|\mathbf{p} - \mathbf{q}\|_\infty \leq 1$ ). It is well known that, to deal with topological paradoxes related to the digital version of the Jordan theorem, we generally use in one binary digital image a pair of different adjacency relations, and denote as  $(\alpha, \beta)$  where  $\alpha$  and  $\beta$  are adjacency relations for foreground (black) and background (white) pixels respectively. In 2D, we consider in particular  $(\alpha, \beta) = (4, 8)$  or  $(8, 4)$ .

In the graph-based framework of digital topology, the concept of *simple point* [9,10] (see Fig. 2) relies on the local notion of adjacency and on the induced global notion of connectedness. The simple points provide a way to characterise the preservation of topological properties in a (binary) image during its transformation. Practically, a pixel  $\mathbf{x} \in S$  of an image  $I : S \rightarrow F$  is *simple* if its binary value can be switched without modifying the topological properties of  $I$ . In particular, the simplicity of a pixel can be tested, in constant time, by only studying its  $3 \times 3$  neighbourhood [9]. We will say that two images  $I$  and  $I'$  are *simple-equivalent* [13] if  $I'$  is obtained from  $I$  by iteratively modifying (successive) simple points. Thus  $I$  and  $I'$  present the same homotopy type.

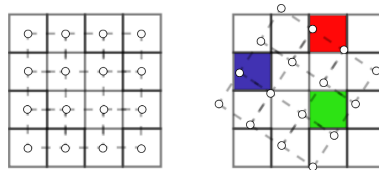
## 3 Discrete Rigid Transformations: Topological Issues

A (continuous) rigid transformation  $\mathcal{T}$  establishes a bijection from  $\mathbb{R}^2$  to itself. By opposition, due to the digitisation process  $D$  (see Eq. (2)), a discrete rigid transformation  $T$  is, most of the time, not a bijection from  $\mathbb{Z}^2$  to itself.

It is plain that for any three distinct pixels  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \in \mathbb{Z}^2$ , we have  $\max_{i,j \in \{1,2,3\}} \{\|\mathbf{x}_i - \mathbf{x}_j\|_2\} \geq \sqrt{2}$ . This leads to the following definition that



**Fig. 2.** Examples of simple points  $(x, y)$  and non-simple points  $(z, t)$ . Modifying the value of  $z$  would merge two black connected components, while modifying the value of  $t$  would create a white connected component. In both cases, the homotopy type of the image would be modified.



**Fig. 3.** Left: a digital image support and the grid modeling its discrete structure. Right: examples of a null pixel (in green), a single pixel (in blue) and a double pixel (in red) with respect to a discrete rigid transformation.

enables to characterise the status of a pixel; there are only three possibilities, as illustrated in Fig. 3.

**Definition 1.** For a pixel  $x \in \mathbb{Z}^2$  and a given DRT  $T$ , let  $M(x) = \{y \in \mathbb{Z}^2 \mid T(y) = x\}$ .

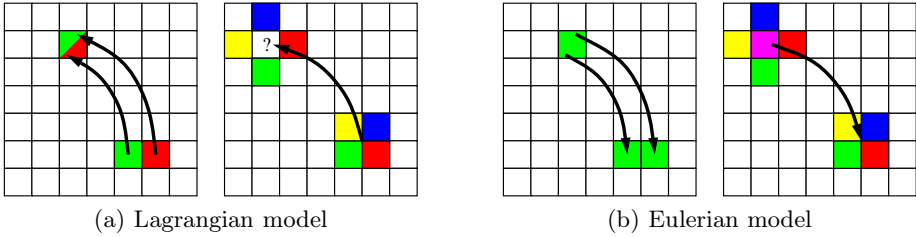
- If  $|M(x)| = 0$ , we say that  $x$  is a null pixel.
- If  $|M(x)| = 1$ , we say that  $x$  is a single pixel.
- If  $|M(x)| = 2$ , we say that  $x$  is a double pixel.

Similar notions for the case of discrete rotations can be found in [5,6].

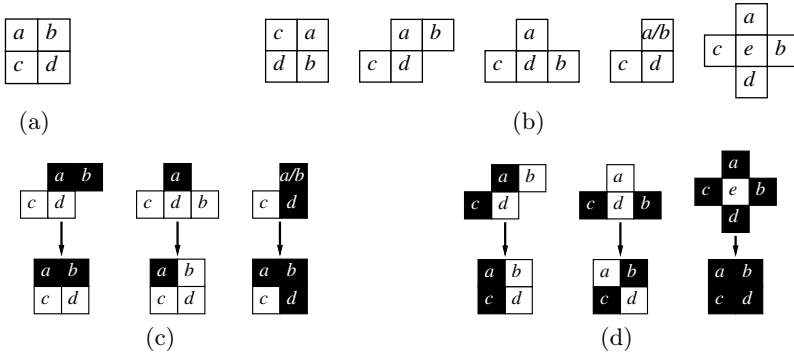
In particular, a discrete rigid transformation  $T$  behaves like a bijection for single pixels. However the possible existence of null (resp. double) pixels may forbid  $T$  to be a surjection (resp. an injection). Null and double pixels thus raise topological issues in both Lagrangian and Eulerian transformation models (see Sec. 3.1). In addition to these “cardinality-based” issues, supplementary topological problems are induced by the alteration of adjacency relations between pixels (see Sec. 3.2).

### 3.1 Transformation Models

Two standard transformation models can be considered: the Lagrangian (or forwards) and the Eulerian (or backwards) models. The Lagrangian model consists of observing  $\mathcal{T}(x)$  for every pixel  $x$  in the initial space, while the Eulerian model consists of observing  $\mathcal{T}^{-1}(y)$  for every pixel  $y$  in the transformed space. These models are equivalent in  $\mathbb{R}^2$ , since  $\mathcal{T}$  is bijective. In  $\mathbb{Z}^2$ , they are however generally distinct, since  $T$  and  $T^{-1}$  may not be inverse functions.



**Fig. 4.** The interpretations of double pixels (left figures) and null pixels (right figures) in the context of discrete rigid transformations for (a) the Lagrangian and (b) the Eulerian models



**Fig. 5.** (a) A  $2 \times 2$  pixel sample with values  $a, b, c, d$ . (b) Local pixel configurations (up to rotations and symmetries) leading to the sample (a) when applying a discrete rigid transformation. (c) Examples of transformations in which the sample preserves the topology of local pixel configurations. (d) Examples of transformations in which the sample provokes a topological alteration.

Depending on each model, null and double pixels lead to different interpretations. In the Lagrangian model (see Fig. 4(a)), a double pixel in the transformed space may receive two different pixel values, and a null pixel receive no pixel value. While this may be conveniently handled in the case of binary images (which can be considered in a set-based paradigm), it can lead to correctness and completeness issues in the case of multivalued images. In the Eulerian model (see Fig. 4(b)), a double pixel of the initial space may transfer its value to two pixels of the transformed space, while the value of a null pixel will be lost.

In this preliminary work, we consider the Eulerian model which enables us to focus on the topological issues raised by the alteration of adjacency relations (see Sec. 3.2), and as the Lagrangian model is fraught with additional difficulties (see Sec. 7). For the sake of readability, our study is carried out in the context of binary images, but the introduced methodology remains valid in the case of multivalued images (see Sec. 7).

### 3.2 Adjacency Alterations

In order to illustrate the topological issues raised by the alterations of adjacency relations during discrete rigid transformations, let us consider a  $2 \times 2$  pixel sample of the transformed space (see Fig. 5(a)). Such a sample is composed of pixels of values  $a$ ,  $b$ ,  $c$  and  $d$ , and all the possible local pixel configurations of the initial space from which the sample is generated (see Fig. 5(b)). Despite local adjacency alterations between pixels, the global topology of the sample may sometimes be preserved (see Fig. 5(c)). Unfortunately, such local alterations may also lead to topological alterations in the sample (see Fig. 5(d)), and further in the whole image possibly. In the next section, we propose an algorithm enabling the detection of potential topological changes during a DRT. On the contrary, this algorithm can be used to guarantee the topological invariance between an image and all of its transformed ones. This algorithm is based on (i) the recently introduced notion of DRT graph [7,8], and (ii) the classical notion of simple point [9,10]. The first notion provides a way to exhaustively explore the space of transformed (sub)images while the second provides information on the possible topological modifications when performing such an exploration.

## 4 Mathematical Tools for Topological Verification of Images under Rigid Transformations

### 4.1 Discrete Rigid Transformation Graph

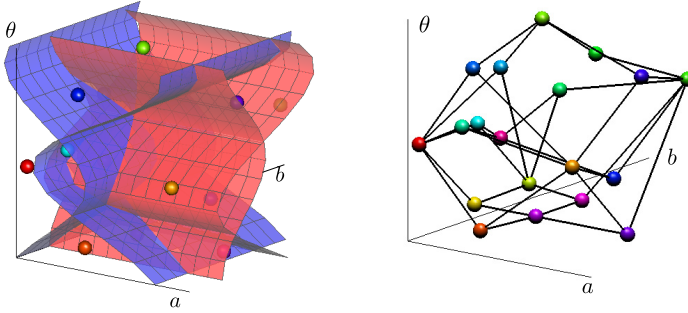
In opposition to rigid transformations in  $\mathbb{R}^2$ , discrete rigid transformations (DRTs) are not continuously defined w.r.t. the parameters  $a$ ,  $b$  and  $\theta$ . In particular, the parameter space  $(a, b, \theta) \in \mathbb{R}^3$  is divided into 3D open cells, in each of which the functions  $T_{ab\theta}$  are equal, while the 2D surfaces bounding these open cells represent to discontinuities of the DRTs, induced by the digitisation process (see Eq. (2)). In fact, each 2D surface is corresponded to an elementary modification of one pixel.

From a theoretical point of view, each 3D open cell can be seen as the equivalent class of the rigid transformations  $\mathcal{T}$  that leads to a same DRT  $T = D \circ \mathcal{T}$  whose boundaries are the 2D surfaces. By mapping any 3D cell onto a 0D point and any 2D surface onto a 1D edge, the combinatorial structure of the parameter space can be modeled in a dual way by a connected graph, as illustrated in Fig. 6. In particular, each 0D point corresponds to a transformed image generated by the associated  $T$  and an 1D edge between two cells indicates that the two associated images differ at exactly one pixel.

**Definition 2 (DRT graph [7]).** A DRT graph  $G = (V, E)$  is defined such that:

- any vertex  $v \in V$  models a 3D open cell and associates to a transformed image;
- any (labeled) edge  $e = (v, w, (\mathbf{p}, \mathbf{p}')) \in E$  models a 2D discontinuity surface between the transformed images corresponding to the DRTs  $v$  and  $w$  which differ at the single pixel  $\mathbf{p}'$ , and  $\mathbf{p}$  is the pixel corresponding to  $\mathbf{p}'$  in the original image.





**Fig. 6.** A part of the parameter space subdivided by four 2D surfaces corresponding to the discontinuities of DRTs (left), and the associated part of the DRT graph (right)

The label  $(\mathbf{p}, \mathbf{p}')$  on each edge is –implicitly– associated to a function indicating the value modification of the pixel  $\mathbf{p}'$  that differs between the transformed image associated to DRTs  $v$  and  $w$ . More precisely, the value of  $\mathbf{p}'$  at the vertex  $v$  is defined by  $I_v(\mathbf{p}') = I(\mathbf{p})$  where  $I : S \rightarrow F$  is the original image function. After the elementary rigid motion at  $e$ , we then obtain a new image  $I_w$  by simply changing the pixel value at  $\mathbf{p}'$  as  $I_w(\mathbf{p}') = I(\mathbf{p} + \delta)$  where  $\delta = (\pm 1, 0)$  or  $(0, \pm 1)$ . Note that  $\delta$  corresponds to an *elementary motion*, i.e., a smallest pixel movement, that changes either  $x$ - or  $y$ -coordinate by 1.

It was proved in [7] that the DRT graph associated to a digital image of size  $N^2$  has a space complexity of  $\mathcal{O}(N^9)$  (and can be built with a similar time complexity [7,8]). Note that the structure of the DRT graph depends only on the support of the given images, but not on their pixel values. By construction, the DRT graph provides all the transformed images of a given image  $I$ . In particular, these transformed images can be generated by progressively and exhaustively scanning the DRT graph.

**Remark 3.** Let  $I : S \rightarrow F$ , and  $G = (V, E)$  be its DRT graph. For each edge  $e = (v, w, (\mathbf{p}, \mathbf{p}'))$  (i.e., each elementary modification of a pixel  $\mathbf{p}' \in S$ ), two cases can occur:

- (i)  $I_v(\mathbf{p}') = I_w(\mathbf{p}')$ , i.e., the images of  $I$  by the DRTs  $v$  and  $w$  are equal ( $I_v = I_w$ );
- (ii)  $I_v(\mathbf{p}') \neq I_w(\mathbf{p}')$ , i.e.,  $I_v \neq I_w$ .

In the (considered) case of binary images, the value of  $\mathbf{p}'$  may then be flipped from white to black (or vice versa), and this may constitute the only modification between the transformed images of  $I$  by the DRTs  $v$  and  $w$ .

## 4.2 Topological Analysis of Binary Images from DRT Graphs

From a DRT graph, one can generate exhaustively all the DRT images of an image  $I$ . Moreover, from Rem. 3, we know that this can be done by modifying (at most) one pixel value between two successive transformed images. In the case of binary images, it is actually possible to check if such a local modification involves a simple point.

**Algorithm 1.** Construction of simple-equivalent DRT images**Input:** A DRT graph  $G = (V, E)$ ; the vertex  $u \in V$  associated to  $I$ .**Output:** A connected partial subgraph  $G'' = (V', E'')$  of vertices simple-equivalent to  $u/I$ .

---

```

1  $V' \leftarrow \{u\}; E'' \leftarrow \emptyset; \mathcal{S} \leftarrow \{u\}; U \leftarrow V;$ 
2 while  $\mathcal{S} \neq \emptyset$  do
3   Let  $v \in \mathcal{S}; \mathcal{S} \leftarrow \mathcal{S} \setminus \{v\};$ 
4   if  $v \in U$  then
5      $U \leftarrow U \setminus \{v\};$ 
6     foreach  $e = (v, w, (\mathbf{p}, \mathbf{p}')) \in E$  such that  $w \in U$  do
7       if  $\mathbf{p}'$  is a simple point then  $V' \leftarrow V' \cup \{w\}; E'' \leftarrow E'' \cup \{e\};$ 
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{w\};$ 

```

---

Practically, the edges of a DRT graph  $G = (V, E)$  can then be classified in two categories: those that do not modify the topology of the transformed images (*i.e.*, the edges that correspond to case (i) in Rem. 3, as well as those that correspond to case (ii) for which  $\mathbf{p}'$  is a simple point); and those that modify this topology (*i.e.*, the edges that correspond to case (ii) in Rem. 3 for which  $\mathbf{p}'$  is not simple).

The partial graph  $G' = (V, E')$  is obtained by maintaining only the edges  $E' \subseteq E$  of the first category.  $G'$  is composed of connected components of vertices whose associated transformed images are simple-equivalent (see Sec. 2.2), and thus have the same homotopy type. In particular, the connected component contains the vertex  $u$  corresponding to the initial image  $I$ , as well as those corresponding to transformed images obtained from  $I$  by elementary motion sequences which are topology-preserving. This specific set of vertices can be straightforwardly computed by using a standard spanning-tree algorithm, initialized from the seed vertex  $u$  (see Alg. 1).

**Remark 4.** *The connected component of  $G'$  that contains  $u$  may constitute only a strict subset of the vertices/transformed images that are simple-equivalent to  $u/I$ . Indeed, the edges of the DRT graph  $G$  only model the local modifications associated to DRTs. In particular, there may exist other series of local modifications relying on simple points but not modeled in the DRT graph. In other words, the analysis of the DRT graph provides sufficient (but not necessary) conditions to guarantee homotopy-type preservation.*

In the case where  $V' = V$  (see Alg. 1), *i.e.*, when all the vertices of the DRT can be reached from  $u$  by a sequence of edges involving only simple points, the algorithm successfully detects –as a side effect– an image  $I$  that is actually topologically invariant under *any* DRTs. The algorithmic cost of this algorithm is directly linked to the size of the DRT graph, that is  $\mathcal{O}(N^9)$ . This algorithmic complexity is indeed reached in the worst cases. In the next section, we show that this problem can however be decomposed spatially, thus leading to a practical, lower complexity algorithm.

## 5 Local Evaluation of Topological Invariance under DRTs

In the previous section, we have proposed to explore the whole DRT graph of an image  $I$  in order to check its potential topological invariance under DRTs. For each edge  $e = (v, w, (\mathbf{p}, \mathbf{p}'))$  of the DRT graph, we verify that the pixel  $\mathbf{p}'$  whose value is modified at this edge is actually a simple point for the transformed images  $I_v$  and  $I_w$ . This test is performed locally, more precisely in the  $3 \times 3$  neighbourhood centered on  $\mathbf{p}'$  in the transformed space.

We now take advantage of the local nature of these tests to develop a spatial decomposition strategy that will lead to a *local* version of the previously proposed *global* method. To this end, we first need to introduce basic notions and properties related to the influence of DRTs on pixel neighbourhoods.

### 5.1 Neighbourhoods and DRTs

Let  $\mathbf{p} \in \mathbb{Z}^2$  be a pixel. We define the neighbourhoods of  $\mathbf{p}$  as follows:  $N_8(\mathbf{p}) = \{\mathbf{q} \in \mathbb{Z}^2 \mid \|\mathbf{q} - \mathbf{p}\|_2 < 2\}$ ;  $N_{20}(\mathbf{p}) = \{\mathbf{q} \in \mathbb{Z}^2 \mid \|\mathbf{q} - \mathbf{p}\|_2 < 2\sqrt{2}\}$ . The first  $3 \times 3$  neighbourhood is classically used in digital geometry and topology. The second corresponds to a  $5 \times 5$  square from which the 4 extremal corner pixels have been removed. We provide the following property where we consider any arbitrary DRT  $T : \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$ .

**Property 5.** *Let  $\mathbf{p} \in \mathbb{Z}^2$  and  $\mathbf{q} \in N_8(\mathbf{p})$ . We then have  $T^{-1}(\mathbf{q}) \in N_{20}(T^{-1}(\mathbf{p}))$ .*

**Proof.** This property derives from the above definitions of the neighbourhoods, and from the fact that a DRT  $T$  (due to the digitization induced by  $D$ , see Eq. (2)) implies a possible (strict) majoration of  $\sqrt{2}$  for the distance between transformed points, w.r.t. its associated rigid transformation  $\mathcal{T}$ . ■

### 5.2 A Local Approach for Topological Analysis

As stated above, a DRT graph models *all* the rigid transformations of a given digital image  $I$ . Despite the fact that the space of these transformations is actually infinite, the DRT graph is defined as a finite structure. This restriction can be made without loss of correctness/completeness by considering translation invariance. Indeed, a rigid transformation is defined as a composition of a rotation and a translation (Eq. (1)). In particular, a rigid transformation  $\mathcal{T} = \mathbf{t} \circ \mathbf{r}$ , composed of a rotation  $\mathbf{r}$  around the origin and a translation  $\mathbf{t} \in \mathbb{R}^2$ , can be also expressed as  $\mathcal{T} = \mathbf{t} \circ \mathbf{t}'^{-1} \circ \mathbf{r}' \circ \mathbf{t}'$ , where  $\mathbf{t}'$  is the translation by a vector  $\mathbf{p} \in \mathbb{Z}^2$ , and  $\mathbf{r}'$  is the rotation around  $\mathbf{p}$ . Such a translation in  $\mathbb{Z}^2$  (that induces no topological modification, since the whole image is translated), allows any pixel  $\mathbf{p}$  of  $S$  of the image  $I$  to be considered as the origin of  $S$ .

Let come back to the DRT graph  $G = (V, E)$  considered until now, and involved in the global process defined in the previous section. We now focus on an edge  $e = (v, w, (\mathbf{p}, \mathbf{p}'))$  of  $G$ . Obviously, the vertices  $v$  and  $w$  exist in  $G$ , and also in the (equivalent) DRT graph where we consider  $\mathbf{p}$  as origin. In the later graph, any edge that does not involve in its label a pixel of  $N_{20}(\mathbf{p})$  has no

influence on the topological modifications in  $N_8(\mathbf{p}')$  (see Prop. 5). Without loss of correctness, such topological modifications at  $\mathbf{p}'$  in  $N_8(\mathbf{p}')$  (and thus of  $\mathbf{p}'$  in the whole image) only depends on the part of the DRT graph that corresponds to the restriction of  $I$  to  $N_{20}(\mathbf{p})$ , denoted by  $I_{|N_{20}(\mathbf{p})}$ . Based on these considerations, it appears that if for any pixel  $\mathbf{q}$  in the initial image  $I$ , the restriction  $I_{|N_{20}(\mathbf{q})}$  does not lead to topological modifications under any DRTs, then the same conclusion holds for the whole image  $I$ . In other words, every elementary topological change occurring on the DRT graph  $G$  of  $I$  can be observed locally. Therefore, we need only to verify the topological invariance for every pixel of  $I$  in its neighbourhood  $N_{20}$  in the original binary image.

**Proposition 6.** *Given a binary image  $I : S \rightarrow F$ , for every  $\mathbf{p} \in S$  if  $I_{|N_{20}(\mathbf{p})}$  is a local binary configuration which is topologically invariant under any rigid transformations, then the image  $I$  is topologically invariant.*

We assume that the modified pixel  $\mathbf{p}'$  at each elementary rigid modification of the DRT graph corresponds to the origin  $\mathbf{o}_2$  in the transformed image, and that the corresponding pixel  $\mathbf{p}$  is the origin  $\mathbf{o}_1$  in the original image of size in its  $N_{20}(\mathbf{o}_1)$ . Then, we simply need to construct the DRT graph with  $\mathbf{p}' \in [-1, 1]^2$ ,  $\mathbf{p} \in [-2, 1]^2$  (i.e., the DRT graph of edges labels  $(\mathbf{p}, \mathbf{p}')$  with this constraint), denoted by  $G_{\mathbf{p}} = (V_{\mathbf{p}}, E_{\mathbf{p}})$ . We use Alg. 1 proposed in Sec. 4.2 to verify in  $G_{\mathbf{p}}$  the topological equivalence between two adjacent vertices  $v$  and  $w$  whose edge has the label  $(\mathbf{o}_1, \mathbf{o}_2)$ . If every edge has topologically equivalent vertices, then the center point  $\mathbf{o}_1$  of such a configuration is topologically invariant under any rigid transformations. This approach, in particular, leads to the following consideration: if we study the topological invariance property for all the binary image configurations of  $N_{20}(\mathbf{o}_1)$ , we can identify a family of elementary configurations that authorise topological invariance under DRTs.

From Prop. 6, we propose a look-up-table-based algorithm for characterizing the topologically invariant property of any binary image. More precisely, we generate a set  $P_4$  (resp.  $P_8$ ) which contains only topologically invariant configurations in (4, 8)- (resp. (8, 4)-) adjacent relations. Then we use  $P_4$  and  $P_8$  to verify whether the given image is topologically invariant. The method for building  $P_4$  and  $P_8$  is given in Alg. 2. Let  $C$  be the set of all binary image configurations of size  $N_{20}$ , which is used to build  $P_4$  and  $P_8$ ,  $|C| \leq 2^{20}$ . From Rem. 3 we have  $I_v \neq I_w$  if  $I_v(\mathbf{o}_2) \neq I_w(\mathbf{o}_2)$ , where  $I_v(\mathbf{o}_2) = I(\mathbf{o}_1)$  and  $I_w(\mathbf{o}_2) = I(\mathbf{o}_1 + \delta)$ . We thus need to consider the configurations of  $N_{20}$  whose the central pixel value  $I(\mathbf{o}_1)$  and that of its 4-neighbouring pixel  $I(\mathbf{o}_1 + \delta)$  are different, e.g.,  $I(\mathbf{o}_1) = 1$  and  $I(\mathbf{o}_1 + \delta) = 0$ . Here we set  $\delta = (0, 1)$ , i.e.,  $\mathbf{o}_1 + (1, 0)$  is the right pixel of  $\mathbf{o}_1$ . In other words, the pixel values of  $\mathbf{o}_1$  and its right pixel  $\mathbf{o}_1 + \delta$  are pre-set. Under such conditions,  $|C|$  is reduced to  $2^{18}$ . Thanks to the reflection and rotational symmetries, we can again reduce  $|C|$  to 124 260. Then, we use Alg. 1 proposed in Sec. 4.2 to study the topologically invariant property of configurations in  $C$ . We store in  $P_4$  and  $P_8$  the subset of  $C$  containing *only* the topologically invariant configurations w.r.t. the (4, 8)- and (8, 4)-adjacent relations. Using Alg. 2, we obtain sets of 10 643 and 19 446 topologically invariant configurations in  $P_4$  and  $P_8$  respectively. Fig. 7 shows some elements of  $P_4$  and  $P_8$ .

Based on Prop. 6 and the sets  $P_4$  and  $P_8$ , the algorithm for characterizing the topologically invariant property of a given binary image  $I : S \rightarrow F$  by a local verification of pixels is given in Alg 3. The algorithm scans  $I$  and considers for each pixel  $\mathbf{p} \in S$  its  $N_{20}(\mathbf{p})$  with either  $P_4$  or  $P_8$  depending on the binary value of  $\mathbf{p}$ . Note that  $\bar{I}(\mathbf{p}) = 1 - I(\mathbf{p})$ .

---

**Algorithm 2.** Generation of topologically invariant configuration set  $P_4$  (resp.  $P_8$ )

---

**Input:** The DRT graph  $G_{\mathbf{p}} = (V_{\mathbf{p}}, E_{\mathbf{p}})$  and the set  $C$  of 124 260 binary local configurations of size  $N_{20}$ .

**Output:** The set  $P_4$  (resp.  $P_8$ ).

```

1  $P_4 \leftarrow \emptyset$ ; (resp.  $P_8 \leftarrow \emptyset$ );
2 foreach  $I_C \in C$  do
3    $B \leftarrow TRUE$ ;  $U \leftarrow V_{\mathbf{p}}$ ;  $S \leftarrow \{u\}$  where  $u$  is the vertex associated to  $I_C$  in  $G_{\mathbf{p}}$ ;
4   while  $S \neq \emptyset$  and  $B = TRUE$  do
5     Let  $v \in S$ ;  $S \leftarrow S \setminus \{v\}$ ;
6     if  $v \in U$  then
7        $U \leftarrow U \setminus \{v\}$ ;
8       foreach  $e = (v, w, (\mathbf{p}, \mathbf{p}')) \in E_{\mathbf{p}}$  such that  $w \in U$  do  $S \leftarrow S \cup \{w\}$ ;
9       if  $\exists e = (v, w, (\mathbf{o}_1, \mathbf{o}_2)) \in E_{\mathbf{p}}$  such that  $w \in U$  and  $\mathbf{o}_2$  is not a simple point in its (4, 8) (resp. (8, 4))-adjacency relations then
10         $B \leftarrow FALSE$ ;
11 if  $B = TRUE$  then  $P_4 \leftarrow P_4 \cup \{I_C\}$ ; (resp.  $P_8 \leftarrow P_8 \cup \{I_C\}$ );
```

---



---

**Algorithm 3.** Local verification of the topology invariance of a binary image

---

**Input:** A binary image  $I : S \rightarrow F$  and the sets  $P_4$  and  $P_8$ .

**Output:** Yes if  $I$  is topologically invariant and No otherwise.

```

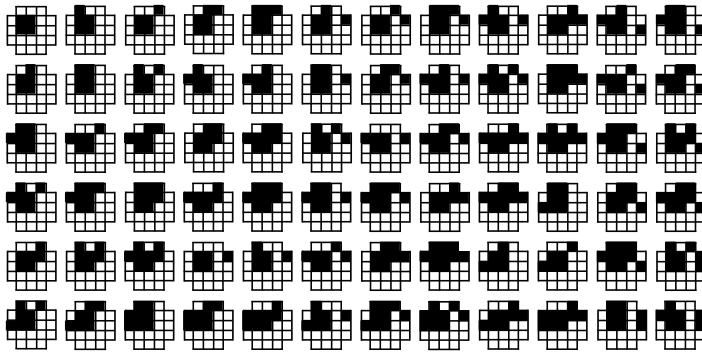
1 foreach  $\mathbf{p} \in S$  do
2   if  $I(\mathbf{p}) = 1$  and  $I_{|N_{20}(\mathbf{p})} \notin P_4$  then return No;
3   if  $I(\mathbf{p}) = 0$  and  $\bar{I}_{|N_{20}(\mathbf{p})} \notin P_8$  then return No;
4 return Yes;
```

---

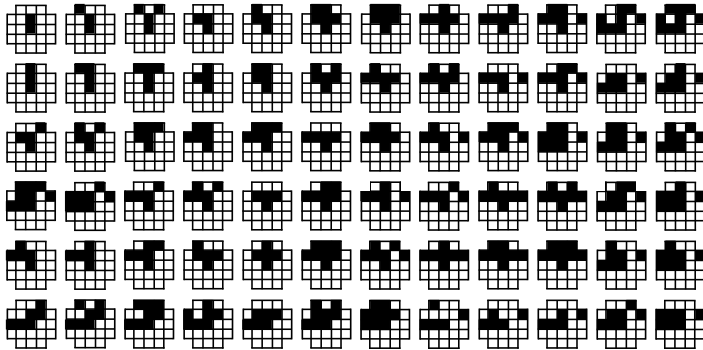
## 6 Experiments

In this section, we illustrate the relevance of our approach by presenting images which have been detected as topology-invariant (see Fig. 8(a-c),(d-f)), or topology-variant (see Fig. 8(g,h), (i,j), (k,l)). Thanks to our LUT-based approach, such a detection can be carried out in linear time w.r.t. the image size.

As mentioned in Rem. 4, we only have a sufficient condition for homotopy-type preservation, so far we do not have a proof for a necessary condition. Nonetheless we have not found any example for which our algorithm fails to characterize its topological invariance.

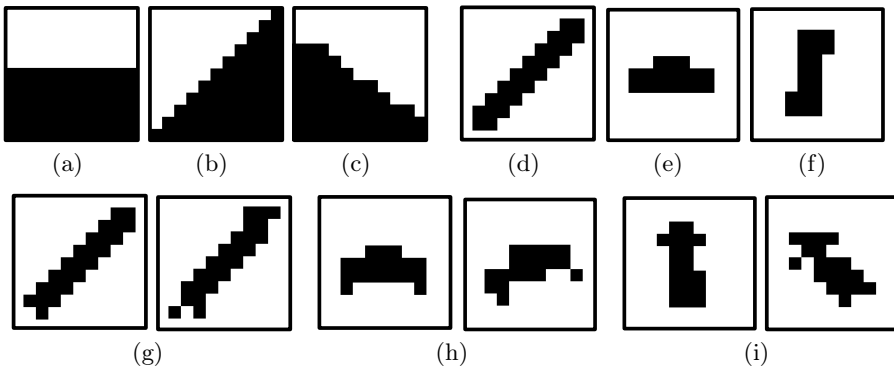


(a)



(b)

**Fig. 7.** Some topologically invariant configurations of  $P_4$  (a) and  $P_8$  (b)



**Fig. 8.** (a-f) Some examples of topology-invariant images. (g-i) Three examples of topology-variant images (left) with their transformed images (right) exhibiting different topologies from their respective original images.

## 7 Conclusion

We have proposed an algorithmic process for determining the topological invariance of digital images under any discrete rigid transformations. This work is based on the recently introduced notion of DRT graph [7,8], which presents a polynomial complexity that generally forbids its practical application on whole images. Nevertheless, DRT graphs have been successfully involved in a preliminary local analysis that finally led to a low complexity methodology, relying on image spatial decomposition.

Beyond its theoretical aspects, this work may contribute to the better understanding of the relationships that exist between geometry and topology in the framework of digital imaging, where both notions are more strongly linked than in continuous spaces.

This study was carried out in the context of binary images. However, it remains relevant whenever a notion of simple point (or more generally a local characterisation of topology preservation) is available. This is verified, for instance, in the context of  $n$ -ary images [14]. On the other hand, only the Eulerian (backwards) model has been considered in this study. In future work, we will extend these results to the case of the Lagrangian (forwards) model. Note that additional difficulties arise in the Lagrangian model, such as double pixels in the transformed space that may receive two different values, and null pixels that do not have any value. The Lagrangian model thus involves a value decision problem for such pixels.

## References

1. Zitová, B., Flusser, J.: Image registration methods: A survey. *Image and Vision Computing* 21, 977–1000 (2003)
2. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. *ACM Computing Surveys* 38, 1–45 (2006)
3. Jacob, M.A., Andres, E.: On discrete rotations. In: *DGCI, Proceedings*, pp. 161–174 (1995)
4. Andres, E.: The Quasi-Shear Rotation. In: Miguet, S., Ubéda, S., Montanvert, A. (eds.) *DGCI 1996. LNCS, vol. 1176*, pp. 307–314. Springer, Heidelberg (1996)
5. Nouvel, B., Rémila, E.: Configurations induced by discrete rotations: Periodicity and quasi-periodicity properties. *DAM* 147, 325–343 (2005)
6. Thibault, Y., Kenmochi, Y., Sugimoto, A.: Computing upper and lower bounds of rotation angles from digital images. *Pattern Recognition* 42, 1708–1717 (2009)
7. Ngo, P., Kenmochi, Y., Passat, N., Talbot, H.: Combinatorial structure of rigid transformations in 2D digital images. To appear in *Computer Vision and Image Understanding*
8. Ngo, P., Kenmochi, Y., Passat, N., Talbot, H.: Combinatorial Properties of 2D Discrete Rigid Transformations under Pixel-Invariance Constraints. In: Barneva, R.P., Brimkov, V.E., Aggarwal, J.K. (eds.) *IWCIA 2012. LNCS, vol. 7655*, pp. 234–248. Springer, Heidelberg (2012)
9. Kong, T.Y., Rosenfeld, A.: Digital topology: Introduction and survey. *Computer Vision Graphics & Image Processing* 48, 357–393 (1989)

10. Couprie, M., Bertrand, G.: New characterizations of simple points in 2D, 3D, and 4D discrete spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31, 637–648 (2009)
11. Khalimsky, E.: Topological structures in computer science. *Journal of Applied Mathematics and Simulation* 1, 25–40 (1987)
12. Kovalevsky, V.A.: Finite topology as applied to image analysis. *Computer Vision, Graphics & Image Processing* 46, 141–161 (1989)
13. Bertrand, G., Couprie, M., Passat, N.: A note on 3-D simple points and simple-equivalence. *Information Processing Letters* 109, 700–704 (2009)
14. Mazo, L., Passat, N., Couprie, M., Ronse, C.: Topology on digital label images. *Journal of Mathematical Imaging and Vision* 44, 254–281 (2012)



# Digital Distances and Integer Sequences

Nicolas Normand<sup>1</sup>, Robin Strand<sup>2</sup>, and Pierre Evenou<sup>1</sup>

<sup>1</sup> LUNAM Université, Université de Nantes, IRCCyN UMR CNRS 6597,  
Polytech Nantes, Rue Christian Pauc, La Chantrerie,  
44306 Nantes Cedex 3, France

<sup>2</sup> Centre for Image Analysis, Uppsala University, Sweden

**Abstract.** In recent years, the theory behind distance functions defined by neighbourhood sequences has been developed in the digital geometry community. A neighbourhood sequence is a sequence of integers, where each element defines a neighbourhood. In this paper, we establish the equivalence between the representation of convex digital disks as an intersection of half-planes ( $\mathcal{H}$ -representation) and the expression of the distance as a maximum of non-decreasing functions.

Both forms can be deduced one from the other by taking advantage of the Lambek-Moser inverse of integer sequences.

Examples with finite sequences, cumulative sequences of periodic sequences and (almost) Beatty sequences are given. In each case, closed-form expressions are given for the distance function and  $\mathcal{H}$ -representation of disks. The results can be used to compute the pair-wise distance between points in constant time and to find optimal parameters for neighbourhood sequences.

## 1 Introduction

A discrete distance function is often defined by using the concept of minimal cost paths obtained by weighted paths and/or neighbourhood sequences [10]. The paths in the traditionally used city-block  $d_4$  and chessboard  $d_8$  distance functions [9] are restricted to the four (and eight) closest neighbours of each grid point in  $\mathbb{Z}^2$ . In [5,2], different weights for closest neighbours and diagonal neighbours are considered giving the weighted distances and in [10,2], the neighbourhood that is allowed in each step is not constant, but defined by a neighbourhood sequence. In [11], both weights and neighbourhood sequences are used to define the distance function. One benefit with the path-generated, discrete, distance functions over the Euclidean distance is when computing point-to-point distances on non-convex domains with the constrained distance transform. The cDT labels each object pixels with the distance to the closest seed pixels, where paths that define the distances are not allowed to cross obstacle pixels. With minimal cost-path distances, the cDT can be computed using standard shortest-path techniques for weighted graphs resulting in a linear time algorithm. A bucket sorting implementation of the Dijkstra's algorithm is used in [12]. See also [11].

A neighbourhood sequence is an integer sequence, but the link between the theory on integer sequences and the theory on discrete distance functions has not been examined in detail yet. In this paper we will apply the theory on integer sequences and use the Lambek-Moser inverse, which is defined for non-decreasing integer sequences, to express the weighted ns-distance. We apply the so-obtained expression to parameter optimization. In [3], neighbourhood sequences on the form of Beatty sequences are considered.

By establishing a link between integer (neighbourhood) sequences and the Lambek-Moser inverse, this paper enables the use of Beatty sequences for distance computations. The inverse of a Beatty sequence can be written in closed form and any element can be computed in constant time. This property is useful for efficient computation (using, for example, wave-front propagation, constrained (geodesic) DT), [11]. Moreover, we give  $\mathcal{H}$ -representation of digital disks, which is useful for, for example, parameter optimization by the link between digital disks and their enclosing polyhedron given in this paper.

## 2 Integer Sequences – The Lambek-Moser Inverse and Beatty Sequences

We denote sequences of integers as  $f = (f(1), f(2), \dots)$ . The Lambek-Moser inverse of the non-decreasing sequence  $f$ , denoted by  $f^\dagger$ , is a non-decreasing sequence of integers defined by [4]:

$$\forall m, n \in \mathbb{N}_+^2, f(m) < n \Leftrightarrow f^\dagger(n) \not\prec m \Leftrightarrow f^\dagger(n) \geq m. \quad (1)$$

In Lambek and Moser's paper,  $f$  and  $f^\dagger$  are only defined for positive indices. However, (1) still holds without altering  $f^\dagger$  if we extend the domain of  $f$  to  $\mathbb{Z}$  with  $f(m) = 0, \forall m \leq 0$  (the same applies to  $f^\dagger$ ):

$$\forall m \in \mathbb{Z}, \forall n \in \mathbb{N}_+, m \leq 0 \text{ or } f(m) < n \Leftrightarrow f^\dagger(n) \geq m.$$

**Proposition 1.**  $f^\dagger(f(m) + 1) + 1$  is the rank of the smallest term greater than  $m$  where  $f$  increases [7,8].

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 [7,8].

A *Beatty sequence* is the sequence obtained by taking the integer parts of the multiples of an irrational constant  $\tau$ :  $(\lfloor \tau \rfloor, \lfloor 2\tau \rfloor, \lfloor 3\tau \rfloor, \dots)$  [1]. Beatty sequences with parameter  $\tau \geq 1$  are non-decreasing. We call *Rational Beatty sequence* the sequence produced with a rational parameter  $\tau$ . Hajdu introduced the use of Beatty sequences in the context of discrete distances [3].

The inverse of the Beatty sequence  $f : m \mapsto \lfloor \tau m \rfloor$  with a scalar  $\tau$ , is  $f^\dagger : n \mapsto \lceil \frac{n}{\tau} - 1 \rceil$  whereas the inverse of  $f : m \mapsto \lfloor \frac{am}{b} + c \rfloor$  is  $f^\dagger : m \mapsto \lfloor \frac{bm}{a} - c - 1 \rfloor$  where  $a, b$  and  $c$  are integers.

### 3 Distance Functions and $\mathcal{H}$ -Representation of Balls

#### 3.1 Discrete Distances

**Definition 1 (Discrete distance and metric).** Consider a function  $d : \mathbb{Z}^n \times \mathbb{Z}^n \rightarrow \mathbb{N}$  and the following properties  $\forall x, y, z \in \mathbb{Z}^n, \forall \lambda \in \mathbb{Z}$ :

1. **positive definiteness**  $d(x, y) \geq 0$  and  $d(x, y) = 0 \Leftrightarrow x = y$ ,
2. **symmetry**  $d(x, y) = d(y, x)$ ,
3. **triangle inequality**  $d(x, z) \leq d(x, y) + d(y, z)$ ,

$d$  is called a distance function, or distance, if it verifies conditions 1 and 2 and a metric with conditions 1 to 3.

**Definition 2 (Closed Ball).** For a given distance function  $d$ , the closed ball  $D$  with radius  $r$  centered in  $c$  is the following set of points of  $\mathbb{Z}^n$ :

$$D(c, r) = \{p : d(c, p) \leq r\} . \tag{2}$$

A series of disks is increasing with respect to set inclusion:

$$\forall r \in \mathbb{N}, D(c, r) \subseteq D(c, r + 1) . \tag{3}$$

Moreover, a discrete distance function is completely described by the sequence of its balls.

$$d(O, p) = \min \{r : p \in D(O, r)\} \tag{4}$$

#### 3.2 $\mathcal{H}$ -Representation of Balls

In this section, we will establish the link between digital disks (discrete polytopes) and  $\mathcal{H}$ -polytopes.

**Definition 3 (Polyhedron).** A convex polyhedron is the intersection of a finite set of half-hyperplanes.

**Definition 4 (Polytope).** A polytope is the convex hull of a finite set of points.

**Theorem 1 (Weyl-Minkowski).** A subset of Euclidean space is a polytope if and only if it is a bounded convex polyhedron.

As a result, a polytope in  $\mathbb{R}^n$  can be represented either as the convex hull of its  $k$  vertices ( $\mathcal{V}$ -representation) or by a set of  $l$  half-planes ( $\mathcal{H}$ -representation):

$$P = \text{conv}(\{p_i\}_{1 \leq i \leq k}) = \left\{ p = \sum_{i=1}^k \alpha_i p_i : \alpha_i \in \mathbb{R}_+ \text{ and } \sum_{i=1}^k \alpha_i = 1 \right\} , \tag{5}$$

$$P = \{x : Ax \leq y\} , \tag{6}$$

where  $A$  is a  $l \times n$  matrix,  $y$  a vector of  $n$  values that we name  $\mathcal{H}$ -coefficients of  $P$ . Given two vectors  $u$  and  $v$ , we denote  $u \leq v$  if and only if  $\forall i, u_i \leq v_i$ .

**Definition 5 (Discrete polytope).** A discrete polytope  $Q$  is the intersection of a polytope  $P$  in  $\mathbb{R}$  with  $\mathbb{Z}$  (Gauss discretization of  $\mathcal{P}$ ).

The minimal parameter representation introduced below is introduced in order to avoid redundancies in the representation.

**Definition 6 (Minimal parameter representation).** A minimal parameter  $\mathcal{H}$ -representation of a discrete polytope  $Q$ , denoted  $\widehat{\mathcal{H}}$ -representation, is a  $\mathcal{H}$ -representation of  $P = \{x : Ax \leq y\}$  such that  $y$  is minimal:

$$P = \{x \in \mathbb{Z}^n : Ax \leq y\} \text{ and } \forall i \in [1..l], \exists x \in P : A_i x = y_i , \quad (7)$$

where  $A_i$  stands for the  $i^{\text{th}}$  line of the matrix  $A$ .

The  $\widehat{\mathcal{H}}$  function, introduced for convenience, gives the minimal parameter vector for a given polytope  $P$ :  $\widehat{\mathcal{H}}(P) = \max\{Ax : x \in P\}$ . As a consequence,  $\{x : Ax \leq \widehat{\mathcal{H}}(P)\}$  is the  $\widehat{\mathcal{H}}$ -representation of  $P = \{x : Ax \leq y\}$ .

**Definition 7 (Convex discrete set).** A Set in  $\mathbb{Z}$  is convex if it is a discrete polytope.

By the construction above, any convex discrete set is given by a ( $\widehat{\mathcal{H}}$ -representation of a) discrete polytope.

Our main result, Theorem 2 below, gives a link between digital disks and intersection of half-planes in  $\mathbb{R}^2$ . The half-planes are given by the Lambek-Moser inverse of the sequences  $f_i$  and the matrix  $A$ . This result will be used as an efficient representation for distance computation and parameter optimization.

**Theorem 2.** The following statements are equivalent:

$$D(O, r) = \{p : A_i p \leq f_i^\dagger(r + 1), \forall i\} \quad (8)$$

$$d(O, p) = \max_i \{f_i(A_i p)\} \quad (9)$$

Where, by convention,  $\forall i, \forall r \leq 0, f_i(r) = 0$

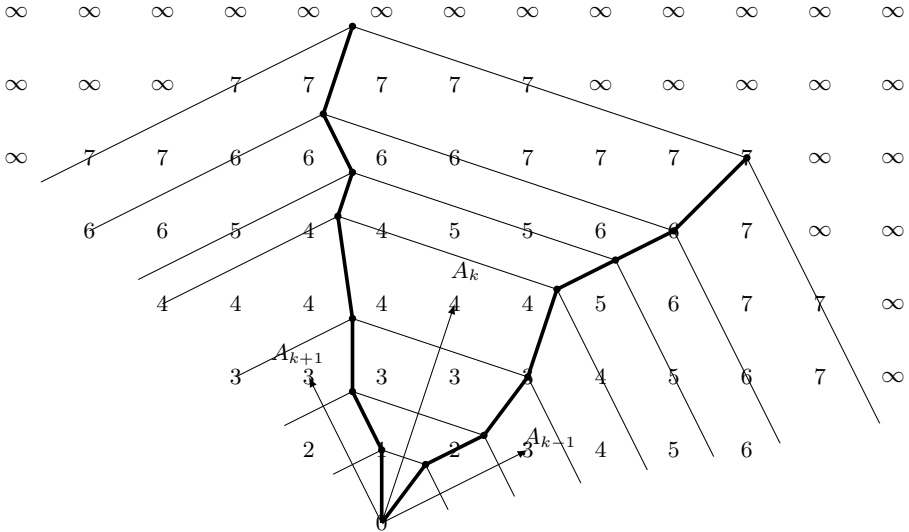
*Proof.* Assume that (9) holds. By definition of a disk:

$$\begin{aligned} D(O, r) &= \{p : d(O, p) \leq r\} \\ &= \{p : \max_i \{f_i(A_i p)\} \leq r\} \\ &= \{p : f_i(A_i p) \leq r, \forall i\} \\ &= \{p : f_i(A_i p) < r + 1, \forall i\} \\ &= \{p : A_i p \leq f_i^\dagger(r + 1), \forall i\} \end{aligned}$$

Conversely,

$$\begin{aligned}
 d(O, p) &= \min\{r : p \in D(O, r)\} \\
 &= \min\{r : A_i p \leq f_i^\dagger(r + 1), \forall i\} \\
 &= \max_i \left\{ \min\{r : A_i p \leq f_i^\dagger(r + 1)\} \right\} \\
 &= \max_i \{ \min\{r : f_i(A_i p) < r + 1\} \} \\
 &= \max_i \{ \min\{r : r \geq f_i(A_i p)\} \} \\
 &= \max_i \{ f_i(A_i p) \} \quad \square
 \end{aligned}$$

Each row  $A_i$  of the matrix  $A$  is a vector normal to a facet of the polytope. The sequence  $f_i$  represents the speed of the polytope growth in direction  $A_i$  which does not need to be uniform as illustrated in Fig. 1.



**Fig. 1.** Illustration of theorem 2. The normal vectors  $A_{k-1}$  to  $A_{k+1}$  are  $(2, 1), (1, 3)$  and  $(-1, 2)$ . The sequences  $f_{k-1}$  to  $f_{k+1}$  are respectively  $(1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 7, \infty)$ ,  $(1, 1, 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 6, 6, 7, 7, 7, \infty)$  and  $(1, 1, 2, 2, 3, 3, 4, 4, 4, 5, 6, 6, 7, 7, \infty)$  and their inverse sequences  $f_{k-1}^\dagger$  to  $f_{k+1}^\dagger$  are  $(0, 2, 4, 6, 8, 10, 12, 15, 16)$ ,  $(0, 3, 5, 8, 12, 14, 16, 20, 24)$  and  $(0, 2, 4, 6, 9, 10, 12, 14, 15)$ . Distance values computed with (9) are given for each discrete point.

## 4 Neighbourhood Sequences and Lambek-Moser Inverse

### 4.1 Weighted Neighbourhood Sequences

We recall some definitions on weighted neighbourhood sequences from [11]. Two grid points  $\mathbf{p}_1 = (x_1, y_1), \mathbf{p}_2 = (x_2, y_2) \in \mathbb{Z}^2$  are  $\rho$ -neighbours,  $\rho \in \{1, 2\}$ , if

$$\begin{aligned} |x_1 - x_2| + |y_1 - y_2| &\leq \rho \text{ and} \\ \max\{|x_1 - x_2|, |y_1 - y_2|\} &= 1. \end{aligned} \tag{10}$$

The points  $\mathbf{p}_1, \mathbf{p}_2$  are *adjacent* if  $\mathbf{p}_1$  and  $\mathbf{p}_2$  are  $\rho$ -neighbours for some  $\rho$ . Two  $\rho$ -neighbours such that the equality in (10) is attained are called *strict*  $\rho$ -neighbours. A ns  $B$  is a sequence  $B = (b(i))_{i=1}^\infty$ , where each  $b(i)$  denotes a neighbourhood 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  $B = (b(1), b(2), \dots, b(l))$ .

We use

$$\mathbf{1}_B(k) = |\{i : b(i) = 1, 1 \leq i \leq k\}| \text{ and } \mathbf{2}_B(k) = |\{i : b(i) = 2, 1 \leq i \leq k\}|.$$

to denote the number of 1:s and 2:s in the ns  $B$  up to position  $k$ .

A *path*, denoted  $\mathcal{P}$ , in a grid 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  $b(i)$ -neighbours. The number of 1-steps and strict 2-steps in a given path  $\mathcal{P}$  is denoted  $\mathbf{1}_\mathcal{P}$  and  $\mathbf{2}_\mathcal{P}$ , respectively.

**Definition 8.** 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  $\alpha$  and  $\beta$  (the *weights*) and a  $B$ -path  $\mathcal{P}$  of length  $n$ , where exactly  $l$  ( $l \leq n$ ) pairs of adjacent grid points in the path are strict 2-neighbours be given. The *cost* of the  $(\alpha, \beta)$ -weighted  $B$ -path  $\mathcal{P}$  is  $(n - l)\alpha + l\beta$ . The  $B$ -path  $\mathcal{P}$  between the points  $\mathbf{p}_0$  and  $\mathbf{p}_n$  is a  $(\alpha, \beta)$ -weighted minimal cost  $B$ -path between the points  $\mathbf{p}_0$  and  $\mathbf{p}_n$  if no other  $(\alpha, \beta)$ -weighted  $B$ -path between the points has lower cost than the  $(\alpha, \beta)$ -weighted  $B$ -path  $\mathcal{P}$ .

**Definition 9.** Given the ns  $B$  and the weights  $\alpha, \beta$ , the weighted ns-distance  $d_{\alpha, \beta}(\mathbf{p}_0, \mathbf{p}_n; B)$  is the cost of (one of) the  $(\alpha, \beta)$ -weighted minimal cost  $B$ -path(s) between the points.

We denote the cumulative sum  $B^\Sigma(k)$  of the ns  $B$  as

$$B^\Sigma(k) = \sum_{l=1}^k B(l) = k + \mathbf{2}_B(k)$$

The following construction gives a non-decreasing sequence of integers:  $f(k) = B^\Sigma(k - 1) + 1 = \mathbf{2}_B(k - 1) + k$ .

*Example 1.* Given the neighbourhood sequence  $B = (1, 2, 1, 2, 2)$ , the cumulative sum is  $B^\Sigma(m) = \lfloor \frac{8}{5}m \rfloor$ , its inverse is  $B^{\Sigma \dagger}(m) = \lceil \frac{5}{8}m - 1 \rceil$ , the non-decreasing sequence  $f$  defined above is  $f(m) = \lfloor \frac{8}{5}m - \frac{3}{5} \rfloor$ , and its inverse is  $f^\dagger(m) = \lceil \frac{5}{8}(m - 1) \rceil$ .

$m$	1	2	3	4	5	6	7	8	9	10	11
$B(m)$	1	2	1	2	2	1	2	1	2	2	1
$B^\Sigma(m)$	1	3	4	6	8	9	11	12	14	16	17
$B^{\Sigma^\dagger}(m)$	0	1	1	2	3	3	4	4	5	6	6
$f(m)$	1	2	4	5	7	9	10	12	13	15	17
$f^\dagger(m)$	0	1	2	2	3	4	4	5	5	6	7

The following formula for weighted ns-distance is given in [11].

$$d(0, (x, y)) = \alpha(2k - x - y) + \beta(x + y - k), \text{ where} \tag{11}$$

$$k = \min\{l : l \geq x + \max\{0, y - 2_B(l)\}\} \tag{12}$$

Proposition 2 below gives an alternative formula for (12). Since the Lambek-Moser inverse  $f^\dagger$  of a Beatty sequence can be written on closed form, it is very efficient to compute.

**Proposition 2.**  $\min\{l : l \geq x + \max\{0, y - 2_B(l)\}\} = \max\{x, f^\dagger(x + y + 1)\}$

*Proof.*

$$\begin{aligned} & \{k = \min\{l : l \geq x + \max\{0, y - 2_B(l)\}\}\} \\ & \left\{ \begin{array}{l} k \geq \max\{x, x + y - 2_B(k)\} \\ k - 1 < \max\{x, x + y - 2_B(k - 1)\} \end{array} \right\} \\ & \left\{ \begin{array}{l} k \geq x \\ k \geq x + y - 2_B(k) \\ k = x \text{ or } k \leq x + y - 2_B(k - 1) \end{array} \right\} \\ & \left\{ \begin{array}{l} k \geq x \\ k + 2_B(k) \geq x + y \\ k = x \text{ or } k + 2_B(k - 1) \leq x + y \end{array} \right\} \\ & \left\{ \begin{array}{l} k \geq x \\ f(k + 1) - 1 \geq x + y \\ k = x \text{ or } f(k) \leq x + y \end{array} \right\} \\ & \left\{ \begin{array}{l} k \geq x \\ k + 1 > f^\dagger(x + y + 1) \\ k = x \text{ or } k \leq f^\dagger(x + y + 1) \end{array} \right\} \\ & \left\{ \begin{array}{l} k \geq x \\ k \geq f^\dagger(x + y + 1) \\ k = x \text{ or } k \leq f^\dagger(x + y + 1) \end{array} \right\} \end{aligned} \tag{13}$$

□

**Corollary 1.** (11) can be rewritten as:

$$d(0, (x, y)) = (2\alpha - \beta) \max\{x, f^\dagger(x + y + 1)\} + (\beta - \alpha)(x + y). \tag{13}$$

*Example 2.* With  $\alpha = 4, \beta = 5$  and  $B$  from Example 1.

$$\begin{aligned} d(0, (x, y)) &= 3 \max \{x, f^\dagger(x + y + 1)\} + x + y \\ &= \max \{4x + y, 3f^\dagger(x + y + 1) + x + y\} \\ &= \max \left\{ f_1 \left( \begin{pmatrix} 4 & 1 \\ x & y \end{pmatrix}^t \right), f_2 \left( \begin{pmatrix} 1 & 1 \\ x & y \end{pmatrix}^t \right) \right\} \end{aligned}$$

as in (9) where  $f_1 : m \mapsto f_1(m) = m$  and  $f_2 : m \mapsto 3f^\dagger(m + 1) + m$ .

$m$	1	2	3	4	5	6	7	8	9	10	11
$f_1(m)$	1	2	3	4	5	6	7	8	9	10	11
$f_1^\dagger(m)$	0	1	2	3	4	5	6	7	8	9	10
$f_2(m)$	4	8	9	13	17	18	22	23	27	31	32
$f_2^\dagger(m)$	0	0	0	0	1	1	1	1	2	3	3

### 4.2 Specific Cases

*Example 3 (Special case I – Weighted distances).*  $B = (2), B^\Sigma = (2, 4, 6, \dots), f = (1, 3, 5, 7, \dots) = 2k - 1, f^\dagger = (0, 1, 1, 2, 2, 3, 3, 4, 4, \dots) = \lfloor \frac{k}{2} \rfloor$  When  $x \geq y \geq 0$ ,  $\max\{x, f^\dagger(x + y + 1)\} = x$ , so (13) becomes:

$$\begin{aligned} d(0, (x, y)) &= (2\alpha - \beta)x + (\beta - \alpha)(x + y) \\ &= \alpha x + (\beta - \alpha)y \end{aligned}$$

which is consistent with [2] and can be written in the form of (9) with the matrix  $A = \begin{pmatrix} \alpha & \beta - \alpha \\ 1 & 1 \end{pmatrix}$  and the function  $f_1 : m \mapsto f_1(m) = m$  which Lambek-Moser inverse is  $f_1^\dagger : m \mapsto m - 1$ . A similar distance formulation for chamfer norms with arbitrary large masks was given in [6, (20)].

*Example 4 (Special case II – ns-distances).* A ns-distance is a special case of wns-distance for which path costs are computed with unitary weights ( $\alpha = \beta = 1$ ). Then (13) becomes:

$$d(0, (x, y)) = \max\{x, f^\dagger(x + y + 1)\}.$$

This can be written in the form of (9) with  $A = \begin{pmatrix} \alpha & \beta - \alpha \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$ ,  $f_1 : m \mapsto m$  and  $f_2 : m \mapsto f^\dagger(m + 1)$ . The Lambek-Moser inverses of  $f_1$  and  $f_2$  are  $f_1^\dagger : m \mapsto m - 1$  and  $f_2^\dagger : m \mapsto f(m) - 1$ . I.e.

$$\begin{aligned} D(0, r) &= \{p : g_i(A_i p) < r + 1, \forall i\} \\ &= \{p : A_i p \leq f_i^\dagger(r + 1), \forall i\} \\ &= \left\{ p : \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} p \leq \begin{pmatrix} f_1^\dagger(r + 1) \\ f_2^\dagger(r + 1) \end{pmatrix} \right\} \end{aligned}$$



$m$	1	2	3	4	5	6	7	8	9	10	11
$B(m)$	1	2	1	2	2	1	2	1	2	2	1
$f^\dagger(m)$	0	1	2	2	3	4	4	5	5	6	7
$f_1(m)$	1	2	3	4	5	6	7	8	9	10	11
$f_1^\dagger(m)$	0	1	2	3	4	5	6	7	8	9	10
$f_2(m)$	1	2	2	3	4	4	5	5	6	7	7
$f_2^\dagger(m)$	0	1	3	4	6	8	9	11	12	14	16

### 5 Optimization

In this section, we will find parameters  $\alpha, \beta, \tau$  that minimize the rotational dependency of the wns-distance. The digital disk obtained by wns is

$$D(0, r) = \{p : A_i p \leq f_i^\dagger(r + 1), \forall i\}$$

Now, we restrict the cumulative neighbourhood sequence  $B^\Sigma$  to rational Beatty sequences, i.e. on the form  $B^\Sigma(m) = \lfloor \tau m \rfloor, 1 \leq \tau \leq 2$ .

$$\begin{aligned} \alpha x_1 + (\beta - \alpha)y_1 &\leq f_1^\dagger(r + 1) \\ x_2 + y_2 &\leq f_2^\dagger(r + 1) \end{aligned}$$

With  $f_1^\dagger(r + 1) = r$  and equality above and  $x = x_1 = x_2$  and  $y = y_1 = y_2$ , we calculate the coordinates of the vertices of the  $\mathcal{H}$ -polytopes:

$$\begin{aligned} x &= \frac{r + (\alpha - \beta)f_2^\dagger(r + 1)}{2\alpha - \beta} \\ y &= \frac{\alpha f_2^\dagger(r + 1) - r}{2\alpha - \beta} \end{aligned}$$

With equality above and  $x = x_1$  and  $y_2 = y_1 = 0$ ,

$$x = \frac{r}{\alpha}$$

By symmetry, the whole polygon is given by the vertices

$$\left( \frac{r + (\alpha - \beta)f_2^\dagger(r + 1)}{2\alpha - \beta}, \frac{\alpha f_2^\dagger(r + 1) - r}{2\alpha - \beta} \right), \left( \frac{r}{\alpha}, 0 \right).$$

**Optimization Procedure.** Loop over integer  $\alpha, \beta$  in some predefined interval and rational  $\tau$  (that defines the Beatty sequence) between 1 and 2.

Find the parameters  $\alpha, \beta, \tau$  that gives the minimum  $P^2A$  for each value up to  $R_{\max}$  that is attained for the specific parameters.

*Example 5.*  $\tau$ : 500 uniform samples between 1 and 2.  $1 \leq \alpha \leq \beta \leq 10$  (and  $2\alpha \leq \beta$ ).

radius	$\tau$	$\alpha$	$\beta$	mean $P^2A/(4\pi)$
$\alpha$	2	$t$	$t$	1.27
$2\alpha$	1.5	$t$	$t$	1.1667
$3\alpha$	2	3	4	1.1207
$4\alpha$	2	7	9	1.0937
$5\alpha$	2	7	9	1.0748
$6\alpha$	2	7	9	1.0658
$7\alpha$	2	7	9	1.0605
$8\alpha$	2	7	9	1.0567
$9\alpha$	1.834	7	9	1.0537
$10\alpha$	1.834	7	9	1.0489

Note that in Example 5, the neighbourhood sequence always start with a 1 due to the definition of  $f$ . In Example 6, we use  $f'(k) = f(k) + 1$  instead, which means that the first element in  $B$  instead is always a 2.

*Example 6.*  $\tau$ : 500 uniform samples between 1 and 2.  $1 \leq \alpha \leq \beta \leq 10$  (and  $2\alpha \leq \beta$ ).  $f'(k) = f(k) + 1$

radius	$\tau$	$\alpha$	$\beta$	mean $P^2A/(4\pi)$
$\alpha$	1	$t$	$t$	1.1312
$2\alpha$	1	$t$	$t$	1.1312
$3\alpha$	2	5	6	1.1081
$4\alpha$	1.6680	4	5	1.1024
$5\alpha$	1.75	7	9	1.0758
$6\alpha$	1.75	7	9	1.0644
$7\alpha$	1.75	7	9	1.0567
$8\alpha$	1.6680	7	9	1.0504
$9\alpha$	1.6680	7	9	1.0458
$10\alpha$	1.6680	7	9	1.0424

## 6 Conclusions

In this paper, we express digital distance functions in terms of integer sequences. Our main result, Theorem 2, gives a link between digital distance functions that can be written on the form (9) and the corresponding digital disks. The obtained expressions are elegant and, most importantly, can be computed efficiently. Since the inverse of Beatty sequences can be computed in constant time, this holds also for distance functions written on the form given in Theorem 2.

We give examples on how the new way of expressing the distance functions can be applied to well-known digital distance functions such as weighted distances and distances based on neighbourhood sequences. The so-obtained formulas are used to find optimal parameters for short neighbourhood sequences.

We also believe that the results presented in this paper has the potential of having large impact on the development of the theory on digital distance functions.

## 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., Evenou, P.: Medial axis lookup table and test neighborhood computation for 3D chamfer norms. *Pattern Recognition* 42(10), 2288–2296 (2009)
7. Normand, N., Strand, R., Evenou, P., Arlicot, A.: Path-Based Distance with Varying Weights and Neighborhood Sequences. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) *DGCI 2011*. LNCS, vol. 6607, pp. 199–210. Springer, Heidelberg (2011)
8. Normand, N., Strand, R., Evenou, P., Arlicot, A.: Minimal-delay distance transform for neighborhood-sequence distances in 2D and 3D. *Computer Vision and Image Understanding* (2013)
9. Rosenfeld, A., Pfaltz, J.L.: Sequential operations in digital picture processing. *Journal of the ACM* 13(4), 471–494 (1966)
10. Rosenfeld, A., Pfaltz, J.L.: Distances functions on digital pictures. *Pattern Recognition* 1(1), 33–61 (1968)
11. Strand, R.: Weighted distances based on neighbourhood sequences. *Pattern Recognition Letters* 28(15), 2029–2036 (2007)
12. Verwer, B.J.H., Verbeek, P.W., Dekker, S.T.: An efficient uniform cost algorithm applied to distance transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11(4), 425–429 (1989)

# The Persistence Space in Multidimensional Persistent Homology

Andrea Cerri<sup>1,3</sup> and Claudia Landi<sup>2,3</sup>

<sup>1</sup> IMATI – CNR, Genova, Italia  
andrea.cerri@ge.imati.cnr.it

<sup>2</sup> DISMI, Università di Modena e Reggio Emilia, Italia  
clandi@unimore.it

<sup>3</sup> ARCES, Università di Bologna, Italia

**Abstract.** Multidimensional persistent modules do not admit a concise representation analogous to that provided by persistence diagrams for real-valued functions. However, there is no obstruction for multidimensional persistent Betti numbers to admit one. Therefore, it is reasonable to look for a generalization of persistence diagrams concerning those properties that are related only to persistent Betti numbers. In this paper, the *persistence space* of a vector-valued continuous function is introduced to generalize the concept of persistence diagram in this sense. Furthermore, it is presented a method to visualize topological features of a shape via persistence spaces. Finally, it is shown that this method is resistant to perturbations of the input data.

## 1 Introduction

Analyzing and interpreting digital images and shapes are challenging issues in computer vision, computer graphics and pattern recognition [24,25]. Topological persistence – including the theory of persistent homology [13] and size theory [17] – has been promisingly linked to the aforementioned research fields [2].

The classical persistence setting is continuous. Data can be modeled as a pair  $(X, f)$ , with  $X$  a topological space and  $f : X \rightarrow \mathbb{R}$  a continuous function called a *filtering function*. The role of  $X$  is to represent the data under study, while  $f$  is a descriptor of some properties which are considered relevant to analyze data. The main idea of persistence is to topologically explore the evolution of the sublevel sets of  $f$  in  $X$ . These sets, being nested by inclusion, produce a filtration of  $X$ . Focusing on the occurrence of important topological events along this filtration – such as the birth and death of connected components, tunnels and voids – it is possible to obtain a global description of data, which can be formalized via an algebraic structure called a *persistence module*. Such information can be encoded in a parameterized version of the Betti numbers, known in the literature as *persistent Betti numbers* [14], a *rank invariant* [6] and – for the 0th homology – a *size function* [26]. The key point is that these descriptors can be represented in a very simple and concise way, by means of multi-sets of points called *persistence*

*diagrams*. Moreover, they are stable with respect to the *bottleneck distance*, thus implying resistance to noise [12].

In concrete applications, where images and shapes are digital, the input data is necessarily discrete. According to the problem at hand, spaces can be modeled by discrete structures such as triangle meshes or cubical complexes. Filtering functions are usually taken to be piecewise-linear. Persistence fits nicely in this discrete framework with none or very little changes. In particular, the case of gray-scale images is treated in [23] and [15].

Thanks to this property, persistence is a viable option for analyzing data from the topological perspective, as shown in a number of concrete problems concerning shape comparison and retrieval [1,10], segmentation [19,20], denoising [11], 3D image simplification [23] and reconstruction [27], visualization [21,22].

A common scenario in applications is to deal with multi-parameter information. An example is given by photometric properties, which are usually taken into account for digital image segmentation. Another instance is the analysis of 4D time-varying CT scans in medical imaging. Further examples can be found in contexts such as computational biology and scientific simulations of natural phenomena. In all these cases, the use of vector-valued filtering functions would enable the study of multi-parameter filtrations, whereas a scalar-valued function only gives a one-parameter filtration. Therefore, Frosini and Mulazzani [18] and Carlsson and Zomorodian [6] proposed *multidimensional persistence* to analyze richer and more complex data. Also in this case the passage from continuous to discrete input data works finely, as shown in [7].

A major issue in multidimensional persistence is that, when filtrations depend on multiple parameters, it is not possible to provide a complete and discrete representation for multidimensional persistence modules analogous to that provided by persistence diagrams for one-dimensional persistence modules [6]. This theoretical obstruction discouraged so far the introduction of a multidimensional analogue of the persistence diagram.

Given the importance of persistence diagrams for the use of persistence in concrete tasks, one can immediately see that the lack of such an analogue is a severe drawback for the actual application of multidimensional persistence. Therefore a natural question we may ask is the following one: In which other sense may we hope to construct a generalization of a persistence diagram for the multidimensional setting?

The persistence diagram is known to satisfy the following important properties [12] (see also [8,16]):

- it can be defined via *multiplicities* obtained from persistent Betti numbers;
- it allows to completely reconstruct persistent Betti numbers;
- the coordinates of its off-diagonal points are homological critical values.

Therefore, it is reasonable to require that a generalization of a persistence diagram for the multidimensional setting satisfies all these properties. We underline that, because of the aforementioned impossibility result in [6], no generalization of a persistence diagram exists that can achieve the goal of representing completely a persistence module, but only its persistent Betti numbers. For this

reason, in this paper we will only study persistent Betti numbers and not persistence modules.

The main contribution of the present work is the introduction of a *persistence space*. We show that it generalizes the notion of a persistence diagram in the aforementioned sense. More precisely, we define a persistence space as a multiset of points defined via multiplicities (Definition 3). In the one-dimensional case it coincides with persistence diagrams. Moreover, it allows for a complete reconstruction of multidimensional persistent Betti numbers (Multidimensional Representation Theorem 2), and the coordinates of its off-diagonal points are multidimensional homological critical values (Theorem 5).

Having established these properties (Section 3), the next step is to use persistence spaces to analyze shapes. The tasks we consider are visualization of a summary of topological information of a shape, and comparison of shapes. Indeed these are the main tasks where persistence diagrams are employed. Therefore, as further contributions of this paper, we show that persistence spaces can be visualized and stably compared.

To the best of our knowledge, so far it was impossible to visualize in a single structure the information contained in multidimensional persistent Betti numbers (although a line-by-line visualization method is given, e.g., in [8]). Our visualization method relies on a projection of points of the persistence space onto a lower dimensional space. Moreover, each point is enriched with the persistence value of the topological feature it represents (color-coded). Our visualization procedure is presented in Section 4.

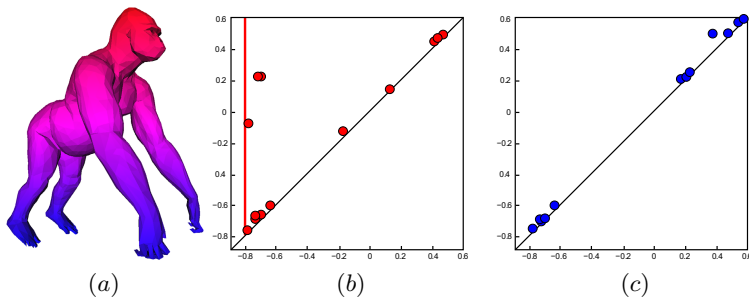
We devote Section 5 to show that the persistence space is resistant to noise. Indeed, persistence spaces can be compared using the multidimensional matching distance introduced in [4]. This comparison turns out to be stable as a simple consequence of the stability of multidimensional persistent Betti numbers [8].

Section 6 concludes the paper with a discussion on the results obtained in this paper and some questions for future research.

## 2 Background

Let us first consider the case when the filtering function  $f$  is real-valued. We can consider the sublevel sets of  $f$  to define a family of subspaces  $X_u = f^{-1}((-\infty, u])$ ,  $u \in \mathbb{R}$ , nested by inclusion, i.e. a *filtration of  $X$* . Homology may be applied to derive some topological information about the filtration of  $X$  induced by  $f$ . The first step is to define persistent homology groups as follows. Given  $u < v \in \mathbb{R}$ , we consider the inclusion of  $X_u$  into  $X_v$ . This inclusion induces a homomorphism of homology groups  $H_k(X_u) \rightarrow H_k(X_v)$  for every  $k \in \mathbb{Z}$ . Its image consists of the  $k$ -homology classes that live at least from  $H_k(X_u)$  to  $H_k(X_v)$  and is called the  *$k$ th persistent homology group of  $(X, f)$  at  $(u, v)$* . If  $X$  satisfies some mild conditions [5] – which will be assumed to hold throughout this paper – this group is finitely generated: Its rank is called a  *$k$ th persistent Betti number of  $(X, f)$* , and is denoted by  $\beta_f(u, v)$  (references to  $X$  and  $k$  are omitted for simplicity).

A simple and compact description for the  $k$ th persistent Betti numbers of  $(X, f)$  is provided by the corresponding persistence diagrams. Figure 1(b) and



**Fig. 1.** (a) A model from the TOSCA dataset [3] with the “height” filtering function color-coded (left), and the corresponding 0th (b) and 1st (c) persistence diagram

Figure 1(c) show respectively the 0th and the 1st persistence diagrams which are obtained from the 0th and the 1st persistent Betti numbers of a surface model – the space  $X$  – filtered by the height function (Fig. 1(a)).

As shown in Fig. 1, persistence diagrams can be seen as multi-sets of points lying in the half-plane  $\Delta^+ = \{(u, v) \in \mathbb{R} \times \mathbb{R} : u < v\}$ . For each point, the  $u$ -coordinate represents the *birth* – in terms of the values of the filtering function – of a topological feature (e.g., connected components in the case of Fig. 1(b), tunnels in the case of Fig. 1(c)), whereas the  $v$ -coordinate represents its *death*. In particular, the red line in Fig. 1(b) can be interpreted as a point at infinity, representing a connected component that *will never die*, i.e. its  $v$ -component is equal to  $+\infty$ . The distance of a point from the diagonal  $\Delta : u = v$  represents the *lifespan* of the associated topological feature, which in turn reflects its importance: points far from the diagonal describe important or global features, i.e. the long-lived ones, whereas points close to the diagonal describe local information such as smaller details and noise. For example, consider the three red points in Fig. 1(b) which are farthest from  $\Delta$ . Together with the red line, they reveal the existence of four meaningful features: The limbs of the gorilla (Fig. 1(b)) born at the four minima of the height function.

A persistence diagram can be formally defined via the notion of *multiplicity* [13,16]. In what follows, the symbol  $\Delta^*$  denotes the set  $\Delta^+ \cup \{(u, \infty) : u \in \mathbb{R}\}$ . Since we assume that the homology degree and the topological space are fixed, we keep omitting any reference to  $k$  and  $X$ .

**Definition 1 (Multiplicity).** *The multiplicity  $\mu_f(u, v)$  of  $(u, v) \in \Delta^+$  is the finite, non-negative number given by*

$$\min_{\substack{\varepsilon > 0 \\ u + \varepsilon < v - \varepsilon}} \beta_f(u + \varepsilon, v - \varepsilon) - \beta_f(u - \varepsilon, v - \varepsilon) - \beta_f(u + \varepsilon, v + \varepsilon) + \beta_f(u - \varepsilon, v + \varepsilon).$$

*The multiplicity  $\mu_f(u, \infty)$  of  $(u, \infty)$  is the finite, non-negative number given by*

$$\min_{\varepsilon > 0, u + \varepsilon < v} \beta_f(u + \varepsilon, v) - \beta_f(u - \varepsilon, v).$$

**Definition 2 (Persistence Diagram).** *The persistence diagram  $\text{Dgm}(f)$  is the multiset of all points  $p \in \Delta^*$  such that  $\mu_f(p) > 0$ , counted with their multiplicity, union the points of  $\Delta$ , counted with infinite multiplicity.*

Definition 2 implies that, given the persistent Betti numbers of  $f$ ,  $\beta_f$ , it is possible to completely and uniquely determine  $\text{Dgm}(f)$ . The converse is also true, as shown by the following result [16], also known as *k-triangle Lemma* [14].

**Theorem 1 (Representation Theorem).** *For every  $(\bar{u}, \bar{v}) \in \Delta^+$  it holds that*

$$\beta_f(\bar{u}, \bar{v}) = \sum_{u \leq \bar{u}, v > \bar{v}} \mu_f(u, v) + \sum_{u \leq \bar{u}} \mu_f(u, \infty).$$

### 2.1 The Multidimensional Setting

If the considered filtering function is vector-valued, i.e.  $f : X \rightarrow \mathbb{R}^n$ , providing the multidimensional analogue of persistent homology groups and Betti numbers is straightforward. For  $u, v \in \mathbb{R}^n$ , with  $u = (u_1, \dots, u_n)$  and  $v = (v_1, \dots, v_n)$ , we say that  $u \prec v$  (resp.  $u \preceq v$ ,  $u \succ v$ ) iff  $u_i < v_i$  (resp.  $u_i \leq v_i$ ,  $u_i > v_i$ ) for every  $i = 1, \dots, n$ . Given  $u \prec v$ , the *multidimensional  $k$ th persistent homology group* of  $(X, f)$  at  $(u, v)$  is defined as the image of the homomorphism  $H_k(X_u) \rightarrow H_k(X_v)$  induced in homology by the inclusion of  $H_k(X_u)$  into  $H_k(X_v)$ . Its rank, still denoted by  $\beta_f(u, v)$ , is called a *multidimensional persistent Betti number*.

What is not straightforward is to generalize Definition 1 and Definition 2. As a consequence, even the multidimensional counterpart of the Representation Theorem 1 cannot be directly deduced from its one-dimensional version. These are actually the main goals of the next section.

## 3 The Persistence Space of a Multi-parameter Filtration

In this section we present the main theoretical results of the paper. Proving most of them is rather technical, and requires a number of intermediate results which, for the sake of clarity, we prefer not to recall here. For more details the reader is referred to the extended version of this work [9].

We start by observing that, in general,  $\beta_f(u, v)$  can be seen as the number of homology classes of cycles “born” no later than  $u$  and “still alive” at  $v$ . Having this in mind, it is easy to figure out that, for  $u', u''$  with  $u' \preceq u''$ , the homology classes of cycles “born” no later than  $u'$  are necessarily not larger in number than the ones “born” no later than  $u''$ . Indeed, it holds that

$$\beta_f(u'', v) - \beta_f(u', v) \geq 0 \tag{1}$$

for every  $v \in \mathbb{R}^n$  with  $u'' \prec v$ . Analogously we can argue that, given  $v', v'' \in \mathbb{R}^n$  with  $u' \preceq u'' \preceq v' \preceq v''$ , the number of homology classes born between  $u'$  and  $u''$  and still alive at  $v'$  is certainly not smaller than the number of those still alive at  $v''$ . More formally,

$$\beta_f(u'', v') - \beta_f(u', v') \geq \beta_f(u'', v'') - \beta_f(u', v''). \tag{2}$$



Let us now set  $\Delta_n^+ = \{(u, v) \in \mathbb{R}^n \times \mathbb{R}^n : u \prec v\}$ . The next step is to recall some properties of the discontinuity points for  $\beta_f$ , which is considered here as a function taking each  $(u, v) \in \Delta^+$  to  $\beta_f(u, v)$ . The next two propositions give some constraints on the presence of discontinuity points for  $\beta_f$ . For every  $\bar{u} \in \mathbb{R}^n$ , we denote by  $\mathbb{R}_\pm^n(\bar{u})$  the subset of  $\mathbb{R}^n$  given by  $\{u \in \mathbb{R}^n : u \prec \bar{u} \vee u \succ \bar{u}\}$ .

**Proposition 1.** *Let  $p = (\bar{u}, \bar{v}) \in \Delta_n^+$ . A real number  $\varepsilon > 0$  exists, such that*

$$W_\varepsilon(p) = \{(u, v) \in \mathbb{R}_\pm^n(\bar{u}) \times \mathbb{R}_\pm^n(\bar{v}) : \|u - \bar{u}\|_\infty < \varepsilon, \|v - \bar{v}\|_\infty < \varepsilon\}$$

*is an open subset of  $\Delta_n^+$ , and does not contain any discontinuity point for  $\beta_f$ .*

**Proposition 2.** *Let  $\bar{u} \in \mathbb{R}^n$ . A real number  $\varepsilon > 0$  exists, such that*

$$V_\varepsilon(\bar{u}) = \{(u, v) \in \mathbb{R}_\pm^n(\bar{u}) \times \mathbb{R}^n : \|u - \bar{u}\|_\infty < \varepsilon, v_i > \frac{1}{\varepsilon}, i = 1, \dots, n\}$$

*is an open subset of  $\Delta_n^+$ , and does not contain any discontinuity point for  $\beta_f$ .*

We can now introduce the multidimensional analogue of Definition 1. For every  $(u, v) \in \Delta_n^+$  and  $e \in \mathbb{R}^n$  with  $e \succ 0$  and  $u + e \prec v - e$ , we consider the number

$$\begin{aligned} \mu_f^e(u, v) &= \beta_f(u + e, v - e) - \beta_f(u - e, v - e) + \\ &\quad - \beta_f(u + e, v + e) + \beta_f(u - e, v + e). \end{aligned} \tag{3}$$

Since we are assuming that the persistent homology groups of  $(X, f)$  are finitely generated, we have that  $\mu_f^e(u, v)$  is an integer number, and by (2) it is non-negative. Once again by (2), if  $\eta \in \mathbb{R}^n$  with  $0 \prec e \preceq \eta$ , then  $\mu_f^e(u, v) \leq \mu_f^\eta(u, v)$ , i.e.  $\mu_f^e(u, v)$  is non-decreasing in  $e$ . Moreover, by Proposition 1 each term in the sum defining  $\mu_f^e(u, v)$  is constant for every  $e \succ 0$  in  $\mathbb{R}^n$  with  $\|e\|_\infty$  sufficiently close to 0. Similarly, the number

$$\beta_f(u + e, v) - \beta_f(u - e, v) \tag{4}$$

is certainly an integer number, non-negative by (1). It is also non-decreasing in  $e$  and non-increasing in  $v$ , as easily implied by (1) and (2), respectively. Moreover, by Proposition 2 each term in (4) is constant for every for  $e \succ 0$  in  $\mathbb{R}^n$  with  $\|e\|_\infty$  sufficiently close to 0, and every  $v \in \mathbb{R}^n$  with  $v_i > \frac{1}{\|e\|_\infty}$  for all  $i = 1, \dots, n$ . The previous remarks justify the following definition.

**Definition 3 (Multiplicity).** *The multiplicity  $\mu_f(u, v)$  of  $(u, v) \in \Delta_n^+$  is the finite, non-negative number defined by setting*

$$\mu_f(u, v) = \min_{\substack{e \succ 0 \\ u+e \prec v-e}} \mu_f^e(u, v). \tag{5}$$

*The multiplicity  $\mu_f(u, \infty)$  of  $(u, \infty)$  is the finite, non-negative number given by*

$$\mu_f(u, \infty) = \min_{e \succ 0, u+e \prec v} \beta_f(u + e, v) - \beta_f(u - e, v). \tag{6}$$

Having extended the notion of multiplicity to a multidimensional setting, the definition of persistence space is now completely analogous to the one of persistence diagram for a real-valued filtering function. Set  $\Delta_n^* = \Delta_n^+ \cup \{(u, \infty) : u \in \mathbb{R}^n\}$  and  $\Delta_n = \partial\Delta_n^+$ .

**Definition 4 (Persistence Space).** *The persistence space  $\text{Spc}(f)$  is the multiset of all points  $p \in \Delta_n^*$  such that  $\mu_f(p) > 0$ , counted with their multiplicity, union the points of  $\Delta_n$ , counted with infinite multiplicity.*

Persistence spaces can be reasonably thought as the analogue – in the case of a multi-parameter filtration – of persistence diagrams. This is due to a number of properties they have in common.

First, it is quite easy to see that, if  $f : X \rightarrow \mathbb{R}^n$  and  $n = 1$ , then Definition 2 and Definition 4 coincide as a simple consequence of the equivalence between Definition 1 and Definition 3.

Second, similarly to the one-dimensional case, a persistence space is completely and uniquely determined by the corresponding persistent Betti numbers. Moreover, even in the multi-parameter situation the converse is true as well, since it is possible to prove the following Multidimensional Representation Theorem. In what follows,  $\langle e \rangle$  denotes the line in  $\mathbb{R}^n$  spanned by  $e$ .

**Theorem 2 (Multidimensional Representation Theorem).** *Let  $(\bar{u}, \bar{v}) \in \Delta_n^+$ . For every  $e \in \mathbb{R}^n$  with  $e \succ 0$ , it holds that*

$$\beta_f(\bar{u}, \bar{v}) = \sum_{\substack{u \prec \bar{u}, v \succ \bar{v} \\ \bar{u}-u, v-\bar{v} \in \langle e \rangle}} \mu_f(u, v) + \sum_{\substack{u \prec \bar{u} \\ \bar{u}-u \in \langle e \rangle}} \mu_f(u, \infty). \tag{7}$$

A further analogy between persistence diagrams and persistence spaces concerns points with positive multiplicity. In both cases, such points can be characterized via the notion of *homological critical value*, introduced in [12] for real-valued functions and in [7] for vector-valued functions.

**Definition 5 (Homological critical value).** *We say that  $u \in \mathbb{R}^n$  is a homological critical value of  $f : X \rightarrow \mathbb{R}^n$  if, for every sufficiently small real value  $\varepsilon > 0$ , there exist  $u', u'' \in \mathbb{R}^n$  such that  $u' \prec u \prec u''$ ,  $\|u' - u\|_\infty \leq \varepsilon$ ,  $\|u'' - u\|_\infty \leq \varepsilon$ , and the homomorphism  $H_k(X_{u'}) \rightarrow H_k(X_{u''})$  induced by inclusion is not an isomorphism for some integer  $k$ .*

It is well known that the coordinates of points in a persistence diagram are homological critical (real) values [12]. The same holds for the points of a persistence space. In fact, the following result can be proved.

**Theorem 3.** *Let  $p$  be a point of  $\text{Spc}(f)$ , with  $\mu_f(p) > 0$ . If  $p = (u, v)$  then both  $u$  and  $v$  are homological critical values for  $f$ . If  $p = (u, \infty)$  then  $u$  is a homological critical value for  $f$ .*

## 4 Visualization of Multidimensional Persistence

One of the main applications of persistence is in visualization and data analysis, mainly due to the fact that the persistence diagram allows us to visualize the appearance and disappearance of topological features in a filtration in a way that is resistant to noise. This is especially important in application areas, where data often come from noisy measurements.

In this section we address the problem of visualizing appearance and disappearance of topological features in a multi-filtration. Our main tool is the persistence space introduced in the previous section.

The idea underlying our visualization method is motivated by the following observations. Recall that, by definition, the persistence space consists of points of  $\Delta_n^* = \Delta_n^+ \cup \Delta_n$ , where  $n$  is the number of parameters in the filtration. In particular, these points are exactly those with a positive multiplicity. For simplicity we can think that this multiplicity is exactly equal to 1, since higher values for multiplicity correspond to non-generic situations. Moreover, intuition suggests that points are arranged on patches of  $2(n-1)$ -dimensional manifolds that intersect each other. For example, for  $n=1$ , we have exactly a set of isolated points in  $\mathbb{R}^2$ , for  $n=2$  we have patches of 2-dimensional surfaces in  $\mathbb{R}^4$ . In general, since  $\Delta_n^* \subseteq \mathbb{R}^n \times \mathbb{R}^n$ , in order to visualize these points, we project them into a lower dimensional space.

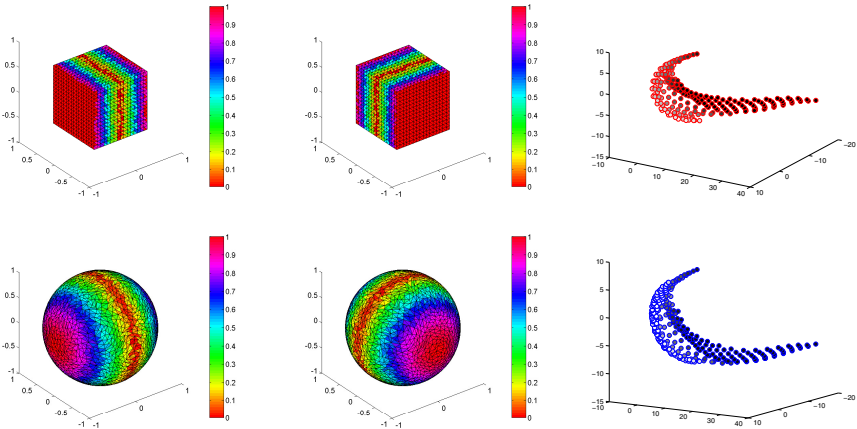
Since each point of the persistence space captures the birth and death of a topological feature along the filtration, how can we visualize its life-time or persistence? In the one-dimensional setting, i.e. for persistence diagrams, this is achieved simply by considering the distance from the diagonal: the larger the distance, the more persistent the feature. In the multidimensional setting, persistence can analogously be defined as the distance from  $\Delta_n$ . However, in the visualization process, when we project onto a lower dimensional space, distances are deformed. Therefore, our idea is to color-code points in the projection according to the persistence of the topological feature they represent.

The result of our method is illustrated in Figure 4 for the case of 1-homology of triangular meshes of a cube and a sphere, respectively. In both cases the considered filtering function on the vertices of the mesh is 2-dimensional and has the first component equal to  $|x|$  (left, color-coded) and the second component equal to  $|y|$  (center, color-coded), and it is interpolated on the other faces. The corresponding persistence spaces are visualized on the right. The darker a point, the more persistent the feature it represents. The projection from  $\mathbb{R}^4$  to  $\mathbb{R}^3$  is obtained using an implementation of Sammon's algorithm.

Having given the main idea of the method, we now explain it step-by-step for a filtering function  $f : X \rightarrow \mathbb{R}^n$ . Basically, the method consists of the following steps:

**Step 1:** Compute a sample of the points  $p = (u, v)$  of the persistence space such that  $u \preceq v$ .

**Step 2:** Compute the persistence of each such point  $(u, v)$  as  $\|v - u\|_\infty$ , and the reciprocal distance – in the max-norm – between these points in  $\mathbb{R}^{2n}$ .



**Fig. 2.** Triangular meshes of a cube and a sphere, endowed with the filtering function  $|x|$  (left),  $|y|$  (center) and a visualization of the 1-homology persistence space for the 2-dimensional function  $(|x|, |y|)$  (right)

- Step 3:** Project these points onto a lower dimensional space by an algorithm that tries to preserve the structure of inter-point distances computed in Step 2 in the lower-dimension projection (e.g., Sammon’s algorithm).
- Step 4:** Plot points coloring them according to the persistence computed in Step 2.

Computations in Step 1 can be accomplished as follows, using the one-dimensional reduction described in [4]. Intuitively, the idea is to consider the set of the so-called *admissible lines*. A line parameterized by  $s$  with equation  $u = sm + b$  is admissible if  $b = (b_1, \dots, b_n)$  is such that  $\sum_{i=1}^n b_i = 0$ , and  $m = (m_1, \dots, m_n)$  is such that  $m_i > 0$  for each  $i$  and  $\sum_{i=1}^n m_i^2 = 1$ . The filtration obtained by sweeping each such line correspond to a persistence diagram: Gluing together all diagrams gives us the persistence space. It follows that, by taking a finite set of admissible lines, we get an approximation of a persistence space. More in details:

- Chose  $k$  admissible lines  $L_1, \dots, L_k$  in  $\mathbb{R}^n$ .
- For  $h = 1, \dots, k$ , consider the line  $L_h$ . Assume its parametric equation is  $u = sm^h + b^h$ . Compute the points  $(s, t) \in \mathbb{R}^2$  of the persistence diagram of the one-dimensional filtration induced by the real-valued function  $F^{L_h}(x) = m_* \cdot \max_{i=1, \dots, n} \left\{ \frac{f_i(x) - b_i^h}{m_i^h} \right\}$ , with  $m_* = \min_i m_i$ .
- For each  $h$  and for each point  $(s, t)$  in the persistence diagram of  $F^{L_h}$ , compute the corresponding point  $(u, v) \in \mathbb{R}^n \times \mathbb{R}^n$  of the persistence space of  $f$  by the formulas  $u = sm^h + b$ ,  $v = tm^h + b$ .

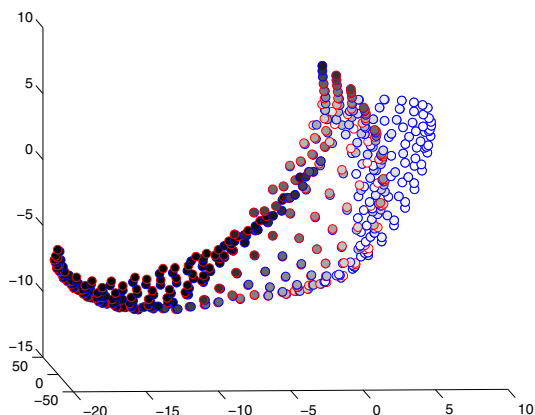
In particular, the above procedure is justified by the correspondence existing between the points in  $\text{Spc}(f)$  and the ones in the persistence diagrams  $\text{Dgm}(F^{L_h})$  [9, Lemma 3.16].

In the next section we shall see that persistence spaces enjoy stability with respect to perturbations of  $f$ . We observe that stability is inherited by the projections obtained according to the proposed method, provided that the point-projection procedure in step 3 preserves the structure of inter-point distances.

### 5 Stable Comparison of Persistence Spaces

A crucial property for applications is the stability of persistence spaces, which means that close functions in the sup-norm should have close persistence spaces in a natural metric.

The reason why our method is stable intuitively is that each admissible line  $L : u = sm + b$  can be associated with a 1-parameter filtration: By sweeping  $L$ , the birth and the death of a topological event occur according to the values of the function  $F^L$  previously defined, whose sublevel sets represent the subspaces of the filtration. It follows that each admissible line  $L$  can be associated to a persistence diagram  $\text{Dgm}(F^L)$ . Moreover, it happens that the collection of persistence diagrams associated with the set  $\mathcal{L}_n$  of all possible admissible lines turns out to be a complete descriptor of the persistent Betti numbers  $\beta_f$ . Therefore, given another filtering function  $g : X \rightarrow \mathbb{R}^n$ , it is possible to define the *multidimensional matching distance*  $D_{\text{match}}$  between  $\beta_f$  and  $\beta_g$  by comparing line by line the associated persistence diagrams via the bottleneck distance  $d_B$  [4]:



**Fig. 3.** Superimposition of the 1-homology persistence spaces of the cube (red points) and the sphere (blue points) of Figure 4

$$D_{\text{match}}(\beta_f, \beta_g) = \sup_{L \in \mathcal{L}_n} d_B(\text{Dgm}(F^L), \text{Dgm}(G^L)).$$

Such a distance is stable with respect to perturbations of the considered vector-valued filtering function: In [8] it has been proved that

$$D_{\text{match}}(\beta_f, \beta_g) \leq \max_{x \in X} \|f(x) - g(x)\|_\infty.$$

Now, we can conclude that, by the Multidimensional Representation Theorem 2, persistence spaces inherit stability from multidimensional persistent Betti numbers.

In Figure 5 stability of persistence spaces is illustrated by displaying the 1-homology persistence spaces of the cube (red points) and the sphere (blue points) of Figure 4 (superimposed). It is clearly visible that points with higher persistence (dark colored) are almost overlapping, while differences are appreciable for points with lower persistence (light colored).

## 6 Conclusions

We have shown that persistence spaces provide a representation of the topological properties of vector-valued functions, and have described how persistence spaces can be visualized. Finally, we have explained how stability of multidimensional persistent Betti numbers implies stability of persistence spaces.

Some questions remain unanswered. Is it possible to further improve this representation by using only a finite set of points, at least in simple cases such as for functions interpolated on vertices of simplicial complexes? Is it possible to explicitly define a bottleneck distance between persistence spaces?

## References

1. Biasotti, S., Bai, X., Bustos, B., Cerri, A., Giorgi, D., Li, L., Mortara, M., Sipiran, I., Zhang, S., Spagnuolo, M.: SHREC'12 Track: Stability on Abstract Shapes, pp. 101–107. Eurographics Association, Cagliari (2012)
2. Biasotti, S., De Floriani, L., Falcidieno, B., Frosini, P., Giorgi, D., Landi, C., Papaleo, L., Spagnuolo, M.: Describing shapes by geometrical-topological properties of real functions. *ACM Comput. Surv.* 40(4), 1–87 (2008)
3. Bronstein, A., Bronstein, M., Kimmel, R.: *Numerical Geometry of Non-Rigid Shapes*, 1st edn. Springer Publishing Company, Incorporated (2008)
4. Cagliari, F., Di Fabio, B., Ferri, M.: One-dimensional reduction of multidimensional persistent homology. *Proc. Amer. Math. Soc.* 138, 3003–3017 (2010)
5. Cagliari, F., Landi, C.: Finiteness of rank invariants of multidimensional persistent homology groups. *Appl. Math. Lett.* 24(4), 516–518 (2011)
6. Carlsson, G., Zomorodian, A.: The theory of multidimensional persistence. *Discr. Comput. Geom.* 42(1), 71–93 (2009)
7. Cavazza, N., Ethier, M., Frosini, P., Kaczynski, T., Landi, C.: Comparison of persistent homologies for vector functions: from continuous to discrete and back (2012), <http://arxiv.org/abs/1201.3217>

8. Cerri, A., Di Fabio, B., Ferri, M., Frosini, P., Landi, C.: Betti numbers in multi-dimensional persistent homology are stable functions. *Math. Method. Appl. Sci.* (in press), doi:10.1002/mma.2704
9. Cerri, A., Landi, C.: Persistence space of vector-valued continuous functions, manuscript, <http://www.dm.unibo.it/~cerri/Publications.html>
10. Chazal, F., Cohen-Steiner, D., Guibas, L.J., Mémoli, F., Oudot, S.: Gromov-Hausdorff stable signatures for shapes using persistence. *Computer Graphics Forum* 28(5), 1393–1403 (2009)
11. Chen, C., Freedman, D.: Topology Noise Removal for Curve and Surface Evolution. In: Menze, B., Langs, G., Tu, Z., Criminisi, A. (eds.) *MICCAI 2010 Workshop MCV. LNCS*, vol. 6533, pp. 31–42. Springer, Heidelberg (2011)
12. Cohen-Steiner, D., Edelsbrunner, H., Harer, J.: Stability of persistence diagrams. *Discr. Comput. Geom.* 37(1), 103–120 (2007)
13. Edelsbrunner, H., Harer, J.: *Computational Topology: An Introduction*. American Mathematical Society (2009)
14. Edelsbrunner, H., Letscher, D., Zomorodian, A.: Topological persistence and simplification. *Discrete Comput. Geom.* 28(4), 511–533 (2002)
15. Edelsbrunner, H., Symonova, O.: The adaptive topology of a digital image. In: 2012 Ninth International Symposium on Voronoi Diagrams in Science and Engineering (ISVD), pp. 41–48 (2012)
16. Frosini, P., Landi, C.: Size functions and formal series. *Appl. Algebra Engrg. Comm. Comput.* 12(4), 327–349 (2001)
17. Frosini, P., Landi, C.: Size theory as a topological tool for computer vision. *Pattern Recogn. and Image Anal.* 9, 596–603 (1999)
18. Frosini, P., Mulazzani, M.: Size homotopy groups for computation of natural size distances. *Bulletin of the Belgian Mathematical Society* 6(3), 455–464 (1999)
19. Letscher, D., Fritts, J.: Image Segmentation Using Topological Persistence. In: Kropatsch, W.G., Kampel, M., Hanbury, A. (eds.) *CAIP 2007. LNCS*, vol. 4673, pp. 587–595. Springer, Heidelberg (2007)
20. Paris, S., Durand, F.: A topological approach to hierarchical segmentation using mean shift. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007*, pp. 1–8 (2007)
21. Pascucci, V., Tricoche, X., Hagen, H., Tierny, J. (eds.): *Topological Methods in Data Analysis and Visualization. Mathematics and Visualization*. Springer (2011)
22. Rieck, B., Mara, H., Leitte, H.: Multivariate data analysis using persistence-based filtering and topological signatures. *IEEE Transactions on Visualization and Computer Graphics* 18, 2382–2391 (2012)
23. Robins, V., Wood, P.J., Sheppard, A.P.: Theory and algorithms for constructing discrete Morse complexes from grayscale digital images. *IEEE Trans. Pattern Anal. Mach. Intell.* 33(8), 1646–1658 (2011)
24. Smeulders, A., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image retrieval at the end of the early years. *IEEE Trans. PAMI* 22(12) (2000)
25. Tangelder, J., Veltkamp, R.: A survey of content-based 3D shape retrieval methods. *Multimedia Tools and Applications* 39(3), 441–471 (2008)
26. Verri, A., Uras, C., Frosini, P., Ferri, M.: On the use of size functions for shape analysis. *Biol. Cybern.* 70, 99–107 (1993)
27. Zheng, Y., Gu, S., Edelsbrunner, H., Tomasi, C., Benfey, P.: Detailed reconstruction of 3D plant root shape. In: Metaxas, D.N., Quan, L., Sanfeliu, A., Gool, L.J.V. (eds.) *ICCV*, pp. 2026–2033. IEEE (2011)

# A Study of Monodromy in the Computation of Multidimensional Persistence

Andrea Cerri<sup>1,4</sup>, Marc Ethier<sup>2</sup>, and Patrizio Frosini<sup>3,4</sup>

<sup>1</sup> IMATI – CNR, Genova, Italia  
andrea.cerri@ge.imati.cnr.it

<sup>2</sup> Département de mathématiques, Université de Sherbrooke,  
Sherbrooke (Québec), Canada  
marc.ethier@usherbrooke.ca

<sup>3</sup> Dipartimento di Matematica, Università di Bologna, Italia  
patrizio.frosini@unibo.it

<sup>4</sup> ARCES, Università di Bologna, Italia

**Abstract.** The computation of multidimensional persistent Betti numbers for a sublevel filtration on a suitable topological space equipped with a  $\mathbb{R}^n$ -valued continuous filtering function can be reduced to the problem of computing persistent Betti numbers for a parameterized family of one-dimensional filtering functions. A notion of continuity for points in persistence diagrams exists over this parameter space excluding a discrete number of so-called *singular* parameter values. We have identified instances of nontrivial monodromy over loops in nonsingular parameter space. In other words, following cornerpoints of the persistence diagrams along nontrivial loops can result in them switching places. This has an important incidence, e.g., in computer-assisted shape recognition, as we believe that new, improved distances between shape signatures can be defined by considering continuous families of matchings between cornerpoints along paths in nonsingular parameter space. Considering that nonhomotopic paths may yield different matchings will therefore be necessary. In this contribution we will discuss theoretical properties of the monodromy in question and give an example of a filtration in which it can be shown to be nontrivial.

**Keywords:** Persistence diagram, topological persistence, multifiltration, shape comparison, shape recognition.

## Introduction

The last few decades have seen an explosion of the amount of data to be processed in many scientific contexts, due in large part to the availability of powerful computing technology. This has led researchers to consider methods to study qualitative information, such as the topological analysis of data, which has been applied, for example, to computer imaging [14] and shape analysis [3].

In this context, *topological persistence* has revealed itself to be an increasingly interesting approach for data analysis and comparison. Indeed, it enables a deep



reduction in the complexity of data, by confining the analysis only to the relevant parts [12,13]. Moreover, persistence allows for a stable description and comparison of data, and ensures invariance under different groups of transformations [3]. The main idea of this theory consists in studying the  $k$ -dimensional holes (components, tunnels, voids . . .) of the sublevel sets of a continuous function, called *filtering function*, varying the level, and using the information gleaned for topological denoising and shape comparison. The relevance of components, tunnels and voids is given by their *persistence*, i.e. the duration of the life of these homological structures if the level is interpreted as time. The more persistent a homological property is, the higher its incidence on shape comparison algorithms, since holes of low persistence are assumed to be the result of noise. A key result shows that if the filtering function is real-valued, then persistent homology can be completely described by a countable collection of points in the real plane, called a *persistence diagram* [10]. The stability of this descriptor with respect to noise makes it important in applications of digital topology and discrete geometry to denoising, where spatial data are only known up to approximation error due to digitization (cf. [6]). Other applications in which persistence has been successfully exploited range from 3D image analysis [1], simplification [17] and reconstruction [18] to image segmentation [16], shape comparison [11] and retrieval [9].

In recent years greater attention has been given to *multidimensional persistence*, i.e. persistent homology for  $\mathbb{R}^n$ -valued filtering functions (e.g. [7]). This extension of the theory is motivated by the fact that data analysis and comparison often involve the examination of properties that are naturally described by vector-valued functions. In computer vision, for example, photometric properties of digital images constitute a standard feature which is taken into account for their segmentation. In point cloud data analysis, the object of study is usually a finite set of samples from some underlying topological space. Each sample is associated with multiple labels, representing several measurements possibly obtained from multiple modalities. Another example is the analysis of 4D time-varying CT scans in medical imaging.

The study of multidimensional persistent homology is proving to be much harder than that of one-dimensional persistent homology. As an example of this difficulty, we recall the lack of a complete and discrete stable descriptor in the case that the filtering function is vector-valued [7]. Fortunately, a method is available to reduce the multidimensional persistent homology of a function  $\varphi : X \rightarrow \mathbb{R}^n$  to the one-dimensional persistent homology of each function in a suitable parameterized family  $\{\varphi_{(m,b)} : X \rightarrow \mathbb{R}\}$  (cf. [2,5]). For each function  $\varphi_{(m,b)}$  a persistence diagram can be obtained, and from the parameterized family of diagrams the multidimensional persistent homology of  $\varphi$  (in the sense of the corresponding persistent Betti numbers) can be recovered.

While this approach has opened a new line of research, it has also brought new problems and questions to the surface. In this work we illustrate that an interesting link exists between the classical concept of monodromy and the persistence diagrams associated with the functions  $\varphi_{(m,b)}$ . In plain words, when we

move in our parameter space  $\{(\mathbf{m}, \mathbf{b})\}$  along a closed path around a so-called singular point, the points in the persistence diagrams may exchange their position. This switching of places generates a monodromy that, besides its theoretical relevance, could be important for defining new distances between multidimensional persistent Betti numbers.

The paper is organized as follows. In Section 1 we recall the definitions and results needed in our exposition. In Section 2 we demonstrate the continuity of the movement of each point in persistence diagrams over paths in parameter space. In Section 3 we describe an example where this continuous movement creates a nontrivial monodromy over persistence diagrams. A section illustrating our conclusions ends the paper.

## 1 Preliminaries

In this section we recall the definitions and results we will be needing in this paper. For the treatment, we refer the reader to [8].

We shall use the following notations:  $\Delta^+$  will be the open set  $\{(u, v) \in \mathbb{R} \times \mathbb{R} : u < v\}$ .  $\Delta$  will represent the diagonal set  $\{(u, v) \in \mathbb{R} \times \mathbb{R} : u = v\}$ . We can further extend  $\Delta^+$  with points at infinity of the kind  $(u, \infty)$ , where  $|u| < \infty$ . Denote this set  $\Delta^*$ . Let us assume that a topological space  $X$  and a continuous function  $\varphi : X \rightarrow \mathbb{R}$  are given. For any  $k \in \mathbb{N}$ , if  $u < v$ , the inclusion map of the sublevel set  $X_u = \{x \in X : \varphi(x) \leq u\}$  into the sublevel set  $X_v = \{x \in X : \varphi(x) \leq v\}$  induces a homomorphism from the  $k$ th homology group of  $X_u$  into the  $k$ th homology group of  $X_v$ . The image of this homomorphism is called the  *$k$ th persistent homology group of  $(X, \varphi)$  at  $(u, v)$* , and is denoted by  $H_k^{(u,v)}(X, \varphi)$ . In other words, the group  $H_k^{(u,v)}(X, \varphi)$  contains all and only the homology classes of  $k$ -cycles born before or at  $u$  and still alive at  $v$ .

In what follows, we shall work with coefficients in a field  $\mathbb{K}$ , so that homology groups are vector spaces. Therefore, they can be completely described by their dimension, leading to the following definition (cf. [13]).

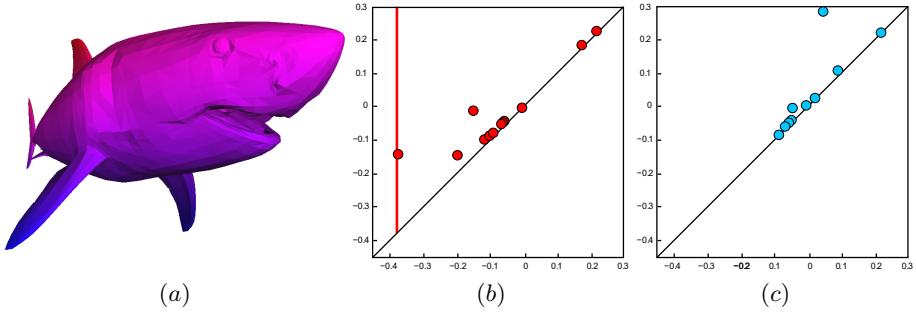
**Definition 1 (Persistent Betti Numbers Function).** *The persistent Betti numbers function of  $\varphi$ , briefly PBN, is the function  $\beta_\varphi : \Delta^+ \rightarrow \mathbb{N} \cup \{\infty\}$  defined by*

$$\beta_\varphi(u, v) = \dim H_k^{(u,v)}(X, \varphi).$$

Throughout the paper, we shall assume that  $X$  is triangulable, implying the finiteness of  $\beta_\varphi$  for all  $(u, v) \in \Delta^+$  [8]. Obviously, for each  $k \in \mathbb{Z}$ , we have different PBNs of  $\varphi$  (which might be denoted  $\beta_{\varphi,k}$ ), but for the sake of notational simplicity we omit adding any reference to  $k$ .

The PBNs of  $\varphi$  can be simply and compactly described by the corresponding *persistence diagrams*. Figure 1(b) and (c) show respectively the 0th and the 1st persistence diagram obtained from the PBNs of the height function defined on a surface model (Fig. 1(a)).

As shown in Figure 1, persistence diagrams can be represented as multisets of points lying in  $\Delta^+$ . For each point, the  $u$ -coordinate denotes the *birth*, in terms



**Fig. 1.** (a) A model from the TOSCA dataset [4] filtered by the “height” function (color-coded, left), and the corresponding 0th (b) and 1st (c) persistence diagrams

of the values of the filtering function, of a topological feature (e.g. connected components in the case of Fig. 1(b), holes in the case of Fig. 1(c)), whereas the  $v$ -coordinate denotes its *death*. In particular, the red line in Fig. 1(b) can be thought as a point at infinity, i.e. a connected component that *will never die*; indeed, its  $v$ -component is equal to  $+\infty$ . The distance of a point from  $\Delta$  can be interpreted as the *lifespan* of the associated topological feature, thus reflecting its importance: points far from the diagonal are associated with important or global features, i.e. long-lived ones, while points close to the diagonal correspond to local information such as smaller details and noise.

Formally, a persistence diagram can be defined via the notion of *multiplicity* [12,15]. Following the convention used for PBNs, any reference to  $k$  will be dropped in the sequel.

**Definition 2 (Multiplicity).** *The multiplicity  $\mu_\varphi(u, v)$  of  $(u, v) \in \Delta^+$  is the finite, non-negative number given by*

$$\min_{\substack{\varepsilon > 0 \\ u + \varepsilon < v - \varepsilon}} \beta_\varphi(u + \varepsilon, v - \varepsilon) - \beta_\varphi(u - \varepsilon, v - \varepsilon) - \beta_\varphi(u + \varepsilon, v + \varepsilon) + \beta_\varphi(u - \varepsilon, v + \varepsilon).$$

*The multiplicity  $\mu_\varphi(u, \infty)$  of  $(u, \infty)$  is the finite, non-negative number given by*

$$\min_{\varepsilon > 0, u + \varepsilon < v} \beta_\varphi(u + \varepsilon, v) - \beta_\varphi(u - \varepsilon, v).$$

**Definition 3 (Persistence Diagram).** *The persistence diagram  $Dgm(\varphi)$  is the multiset of all points  $p \in \Delta^*$  such that  $\mu_\varphi(p) > 0$ , counted with their multiplicity, union the singleton  $\{\Delta\}$ , counted with infinite multiplicity.*

Each point  $p \in \Delta^*$  with positive multiplicity will be called a *cornerpoint*. A cornerpoint  $p$  will be said a *proper cornerpoint* if  $p \in \Delta^+$ , and a *cornerpoint at infinity* if  $p \in \Delta^* \setminus \Delta^+$ .

Persistence diagrams show stability properties with respect to the so-called *bottleneck distance* (a.k.a. *matching distance*). Roughly, small changes in the filtering function induce only small changes in the position of the cornerpoints

which are far from the diagonal in the associated persistence diagram, and possibly produce variations close to the diagonal [8,10]. An intuition of this fact is given in Figure 2. More precisely, we have the following definition:

**Definition 4 (Bottleneck distance).** *Let  $Dgm^1, Dgm^2$  be two persistence diagrams. The bottleneck distance  $d_B(Dgm^1, Dgm^2)$  is defined as*

$$d_B(Dgm^1, Dgm^2) = \min_{\sigma} \max_{p \in Dgm^1} d(p, \sigma(p)),$$

where  $\sigma$  varies among all the bijections between  $Dgm^1$  and  $Dgm^2$  and

$$d((u, v), (u', v')) = \min \left\{ \max \{|u - u'|, |v - v'|\}, \max \left\{ \frac{v - u}{2}, \frac{v' - u'}{2} \right\} \right\} \quad (1)$$

for every  $(u, v), (u', v') \in \Delta^* \cup \{\Delta\}$ .

In practice, the distance  $d$  defined in (1) measures the cost of taking a point  $p$  to a point  $p'$  as the minimum between the cost of moving one point onto the other and the cost of moving both points onto  $\Delta$ . In particular, the matching of a proper cornerpoint  $p$  with  $\Delta$  can be interpreted as the destruction of  $p$ . The stability of persistence diagrams can then be formalized as follows.

**Theorem 1 (Stability Theorem).** *Let  $\varphi, \psi : X \rightarrow \mathbb{R}$  be two filtering functions. It holds that*

$$d_B(Dgm(\varphi), Dgm(\psi)) \leq \|\varphi - \psi\|_{\infty}.$$

We conclude this subsection by noting that if we use Čech homology, persistence diagrams allow the recovery of all information represented in PBNs [8].

### 1.1 Multidimensional Setting

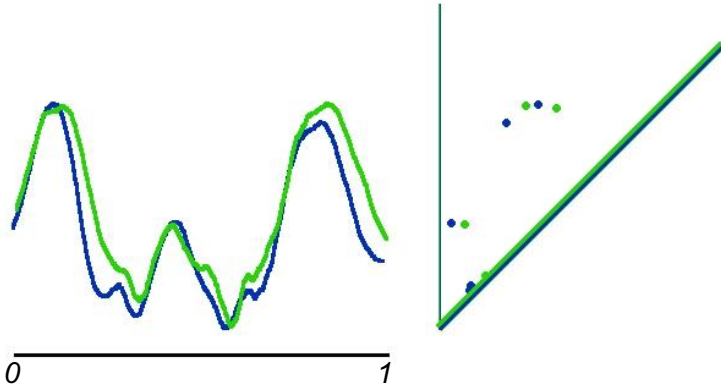
The definition of a persistent Betti numbers function can be easily extended to the case of  $\mathbb{R}^n$ -valued filtering functions [8]. Moreover, it has been proven that the information enclosed in the persistent Betti numbers function of a filtering function  $\varphi = (\varphi_1, \dots, \varphi_n) : X \rightarrow \mathbb{R}^n$  is equivalent to the information represented by the set of persistent Betti numbers functions of the parameterized family  $\{\varphi_{(\mathbf{m}, \mathbf{b})}\}$  of one-dimensional filtering functions defined by setting

$$\varphi_{(\mathbf{m}, \mathbf{b})}(x) = \min_i \{m_i\} \cdot \max_i \left\{ \frac{\varphi_i(x) - b_i}{m_i} \right\}$$

for every  $x \in X$ , varying  $(\mathbf{m}, \mathbf{b})$  in the set of *admissible pairs*

$$Adm_n = \left\{ (\mathbf{m}, \mathbf{b}) \in \mathbb{R}^n \times \mathbb{R}^n : \forall i \ m_i > 0, \sum_i m_i = 1, \sum_i b_i = 0 \right\}.$$

Intuitively, each admissible pair  $(\mathbf{m}, \mathbf{b})$  corresponds to a line of  $\mathbb{R}^n$ , say  $r_{(\mathbf{m}, \mathbf{b})}$ , whose generic point is given by  $u = \tau \mathbf{m} + \mathbf{b}$  with  $\tau \in \mathbb{R}$ . Each such point can be associated with the sublevel set  $X_u$  of  $X$  defined as  $X_u = \{x \in X : \varphi_i(x) \leq u_i, i = 1, \dots, n\}$ . The filtration  $\{X_u\}_{u \in l_{(\mathbf{m}, \mathbf{b})}}$  corresponds to the one associated with  $\varphi_{(\mathbf{m}, \mathbf{b})}$ . Details on this approach to multidimensional persistence can be found in [8].



**Fig. 2.** The change of the filtering function induces a change in the persistence diagram. In this example, the graphs on the left represent two different real-valued filtering functions, defined on the interval  $[0, 1]$ . The corresponding 0th persistence diagrams are displayed on the right.

**Following Proper Cornerpoints.** Obviously, when we change the parameter  $(\mathbf{m}, \mathbf{b})$  in  $Adm_n$ , the cornerpoints of  $Dgm(\varphi_{(\mathbf{m}, \mathbf{b})})$  move in the topological space  $\Delta^* \cup \{\Delta\}$ . The main goal of the present work is to describe some properties of such movements, in the case of proper cornerpoints. To this end, we endow the set  $\Delta^+ \cup \{\Delta\}$  with the metric  $d$  defined in (1).

## 2 Our Main Theorem

We define the pair  $(\mathbf{m}, \mathbf{b}) \in Adm_n$  as *singular for proper cornerpoints* for  $\varphi$  if at least one proper cornerpoint of  $Dgm(\varphi_{(\mathbf{m}, \mathbf{b})})$  has multiplicity strictly greater than 1. Otherwise,  $(\mathbf{m}, \mathbf{b})$  is *regular for proper cornerpoints*. The concept of singularity and regularity would more generally also include the multiplicities of cornerpoints at infinity, but for our purposes here proper cornerpoints suffice. In the sequel, we will therefore use the terms “regular” and “singular” to refer to regularity and singularity with respect to proper cornerpoints. We denote by  $Adm_n^*(\varphi)$  the set of regular pairs of  $\varphi$ . Moreover,  $\varphi$  is said to be *normal* if the set of singular admissible pairs for  $\varphi$  is discrete. We can prove the following result. Let  $I$  be the closed interval  $[0, 1]$ .

**Theorem 2.** *Let  $\varphi : X \rightarrow \mathbb{R}^n$  be a normal filtering function. For every continuous path  $\gamma : I \rightarrow Adm_n^*(\varphi)$  and every proper cornerpoint  $p \in Dgm(\varphi_{\gamma(0)})$ , there exists a continuous function  $c : I \rightarrow \Delta^+ \cup \{\Delta\}$  such that  $c(0) = p$  and  $c(t) \in Dgm(\varphi_{\gamma(t)})$  for all  $t \in I$ . Furthermore, if there is no  $t \in I$  such that  $c(t) = \Delta$ ,  $c$  is the only such continuous function.*

*Proof.* For every  $\delta \geq 0$ , we set  $I^\delta = [0, \delta]$  and consider the following property:

- (\*) a continuous function  $c^\delta : I^\delta \rightarrow \Delta^+ \cup \{\Delta\}$  exists, with  $c^\delta(0) = p$  and  $c^\delta(t) \in \text{Dgm}(\varphi_{\gamma(t)})$  for all  $t \in I^\delta$ .

Define the set  $A = \{\delta \in [0, 1] : \text{property (*) holds}\}$ .  $A$  is non-empty, since  $0 \in A$ . Set  $\bar{\delta} = \sup A$ . We will need to show that  $\bar{\delta} \in A$ . First let  $(\delta_n)$  be a non-decreasing sequence of numbers of  $A$  converging to  $\bar{\delta}$ . Since  $\delta_n \in A$ , for each  $n$  there is a continuous function  $c^{\delta_n} : I^{\delta_n} \rightarrow \Delta^+ \cup \{\Delta\}$  such that  $c^{\delta_n}(0) = p$  and  $c^{\delta_n}(t) \in \text{Dgm}(\varphi_{\gamma(t)})$  for all  $t \in I^{\delta_n}$ . Moreover, we can assume that the following holds:

- (\*\*) if  $m \leq n$ , the restriction of  $c^{\delta_n}$  to the interval  $I^{\delta_m}$  coincides with  $c^{\delta_m}$ .

To clarify this, let us distinguish between two cases. In the first one, we have that  $\Delta \notin c^{\delta_n}(I^{\delta_n})$ , for any  $n$ . Then, if (\*\*) failed, by the Stability Theorem 1 a real value  $\bar{t} \in I^{\delta_m}$  should exist such that  $\text{Dgm}(\varphi_{\gamma(\bar{t})})$  would have a proper cornerpoint of multiplicity strictly greater than 1, which is against our assumption that  $\gamma(t) \in \text{Adm}_n^*(\varphi)$  for all  $t \in I$ . In order to show this, we can set  $\bar{t} = \max\{t \in [0, \delta_m] : c^{\delta_m}(t) = c^{\delta_n}(t)\}$ . The second case is when  $\Delta \in c^{\delta_n}(I^{\delta_n})$  for some indices  $n$ . Let  $m$  be the first index at which this happens, and let  $t' = \min\{t \in [0, \delta_m] : c^{\delta_m}(t) = \Delta\}$ . Then, to ensure that (\*\*) holds, for every  $n \geq m$  we simply modify all the functions  $c^{\delta_n}$  by setting  $c^{\delta_n}(t) = \Delta$  for all  $t \in [t', \delta_n]$ .

Therefore, by (\*\*) we can define a continuous function  $\tilde{c}^\delta : [0, \bar{\delta}] \rightarrow \Delta^+ \cup \{\Delta\}$  such that  $\tilde{c}^\delta(0) = p$  and  $\tilde{c}^\delta(t) \in \text{Dgm}(\varphi_{\gamma(t)})$  for all  $t \in [0, \bar{\delta}]$ . However, to prove that  $\bar{\delta} \in A$  we still need to show that  $\tilde{c}^\delta$  can be continuously extended to the point  $\bar{\delta}$ . The localization of cornerpoints ([8, Prop. 3.8]) implies that, possibly by extracting a convergent subsequence, we can assume that the limit  $\lim_n c^{\delta_n}(\delta_n) = \lim_n \tilde{c}^\delta(\delta_n)$  exists. Once more by the Stability Theorem 1, we have that  $\lim_n \tilde{c}^\delta(\delta_n) \in \text{Dgm}(\varphi_{\gamma(\bar{\delta})})$ . Now the function  $\tilde{c}^\delta$  can be extended to  $\bar{\delta}$  by setting  $\tilde{c}^\delta(\bar{\delta}) = \lim_n \tilde{c}^\delta(\delta_n)$ , to show that  $\bar{\delta} \in A$ .

Last, we prove by contradiction that  $\bar{\delta} = 1$ . Let us suppose that  $\bar{\delta} < 1$ . If  $\tilde{c}^\delta(\bar{\delta}) = \Delta$ , then  $\tilde{c}^\delta$  can be easily extended by setting  $\tilde{c}^\delta(t) = \Delta$  for all  $t \in [\bar{\delta}, 1]$ . Otherwise, by the Stability Theorem 1 and the fact that  $\gamma(\bar{\delta})$  is regular, for any sufficiently small  $\varepsilon > 0$  we can pick a real number  $\eta > 0$  (small enough with respect to  $\varepsilon$ ) such that there is only one proper cornerpoint  $p'(t) \in \text{Dgm}(\varphi_{\gamma(t)})$  with  $d(p'(t), \tilde{c}^\delta(\bar{\delta})) \leq \varepsilon$  for every  $t$  with  $\bar{\delta} \leq t \leq \bar{\delta} + \eta$ . By setting  $\tilde{c}^\delta(t) = p'(t)$  for every such  $t$ , we get a continuous path that extends  $\tilde{c}^\delta$  to the interval  $[0, \bar{\delta} + \eta]$ . In any case, we get a contradiction of our assumption that  $\bar{\delta} = \sup A$ .

We now show the uniqueness of  $c$ , assuming it does not reach  $\Delta$  for any  $t \in I$ . Let  $c, c' : I \rightarrow \Delta^+$  be two continuous paths such that  $c(0) = c'(0) = p$ , and  $c(t), c'(t) \in \text{Dgm}(\varphi_{\gamma(t)})$  for all  $t \in I$ . Denote by  $\bar{t}$  the greatest value such that  $c(t) = c'(t)$  for any  $t$  with  $0 \leq t \leq \bar{t}$ . By the Stability Theorem 1, if  $\bar{t} < 1$  then  $c(\bar{t})$  would be a proper cornerpoint of  $\text{Dgm}(\varphi_{\gamma(\bar{t})})$  having multiplicity strictly

greater than 1. This is against our assumption that  $\gamma(t) \in \text{Adm}_n^*(\varphi)$  for all  $t \in I$ . Therefore  $\bar{t} = 1$ , and our statement is proven.

*Remark 1.* In order to preserve the uniqueness of the function  $c$  even in the cases when it reaches  $\Delta$  for a value  $t \in I$ , what could be done would be to restrict the domain of  $c$  to the subinterval  $[0, \bar{t}]$  such that  $\bar{t} = \sup\{t \in I : \Delta \notin c([0, t])\}$ . Indeed, while there is no way to keep track of proper cornerpoints after they've reached  $\Delta$ , it is also unnecessary to do so, for the simple reason that they then no longer correspond to extant topological properties. This is in contrast to singular admissible pairs, where the definition of  $c$  presents a true ambiguity.

### 3 An Example Illustrating Monodromy in Multidimensional Persistent Homology

Intuition might suggest that a natural correspondence exists between cornerpoints associated to different points in the parameter space  $\text{Adm}_n$ . Example 3 disproves this belief, showing that this correspondence depends on the path followed. Indeed, when following the persistence diagrams  $\text{Dgm}(\varphi_{\gamma(t)})$  along a loop  $\gamma : I \rightarrow \text{Adm}_n^*(\varphi)$ , that is, along a continuous path such that  $\gamma(0) = \gamma(1)$ , nontrivial monodromies may occur if  $\gamma$  is not homotopic to a constant path in  $\text{Adm}_n^*(\varphi)$ . In other words, while  $\text{Dgm}(\varphi_{\gamma(0)}) = \text{Dgm}(\varphi_{\gamma(1)})$ , there may be a  $p \in \text{Dgm}(\varphi_{\gamma(0)})$  such that  $c(1) \neq p$ .

**Example 3 (Nontrivial monodromy).** Consider the function  $\varphi : \mathbb{R}^2 \rightarrow \mathbb{R}^2$  defined on the plane in the following way:  $\varphi_1(x, y) = x$ , and

$$\varphi_2(x, y) = \begin{cases} -x & \text{if } y = 0 \\ -x + 1 & \text{if } y = 1 \\ -2x & \text{if } y = 2 \\ -2x + \frac{5}{4} & \text{if } y = 3 \end{cases},$$

$\varphi_2(x, y)$  then being extended linearly for every  $x$  on the segment joining  $(x, 0)$  with  $(x, 1)$ ,  $(x, 1)$  with  $(x, 2)$ , and  $(x, 2)$  to  $(x, 3)$ . On the half-lines  $\{(x, y) \in \mathbb{R}^2 : y < 0\}$  and  $\{(x, y) \in \mathbb{R}^2 : y > 3\}$ ,  $\varphi_2$  is then being taken with constant slope  $-1$  in the variable  $y$ . The function  $\varphi_2$  is shown plotted in Figure 3. As written the domain of  $\varphi$  is not a compact space, but we can easily obtain a compact domain by considering a suitable subset of  $\mathbb{R}^2$ .

In the case of filtering functions with values in  $\mathbb{R}^2$ ,  $\text{Adm}_2$  is the set of pairs  $(\mathbf{m}, \mathbf{b})$  where  $\mathbf{m} = (a, 1 - a)$  with  $a \in (0, 1)$ , and  $\mathbf{b} = (\beta, -\beta)$  with  $\beta \in \mathbb{R}$ . We may therefore represent an admissible pair as  $(a, \beta) \in (0, 1) \times \mathbb{R}$ . In Figure 4 we can see a “flattened” version of the graph of  $\varphi$ , as if projected on its codomain. It is easily seen that  $\varphi$  admits only one singular admissible pair, that being  $(1/4, 0)$ . Consider a loop  $\gamma$  in  $\text{Adm}_2^*(\varphi)$  moving around this singular pair; for example,  $\gamma(0) = \gamma(1) = (1/4, -\varepsilon)$ ,  $\gamma(1/4) = (1/4 - \varepsilon, \varepsilon)$ ,  $\gamma(1/2) = (1/4, \varepsilon)$ , and  $\gamma(3/4) = (1/4 + \varepsilon, -\varepsilon)$ , with  $\gamma$  linear in  $\text{Adm}_2^*(\varphi)$  between these points. The 0th order persistence diagram  $\text{Dgm}(\varphi_{\gamma(t)})$  is shown for these values of  $t$  in Figure 5;

in each case it has two proper cornerpoints, and this property is in fact true for every  $t \in I$ . If the cornerpoints of  $\text{Dgm}(\varphi_{\gamma(t)})$  are denoted  $p$  and  $q$ , we obtain that  $c_p(1) = q$  and  $c_q(1) = p$ ,  $c_p$  and  $c_q$  being respectively the functions tracking the cornerpoints  $p$  and  $q$ .

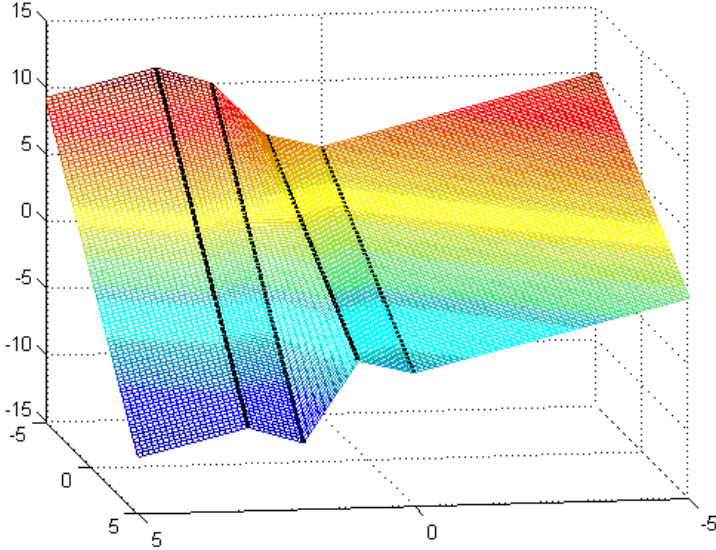


Fig. 3. Function  $\varphi_2$  of Example 3. Depth is  $x$ , width is  $y$

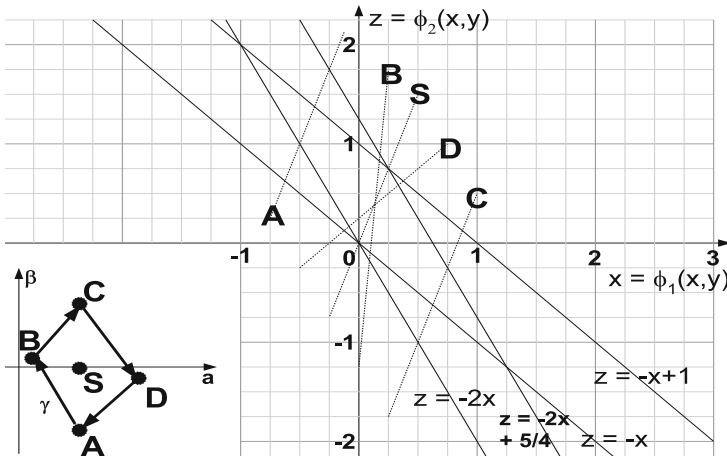
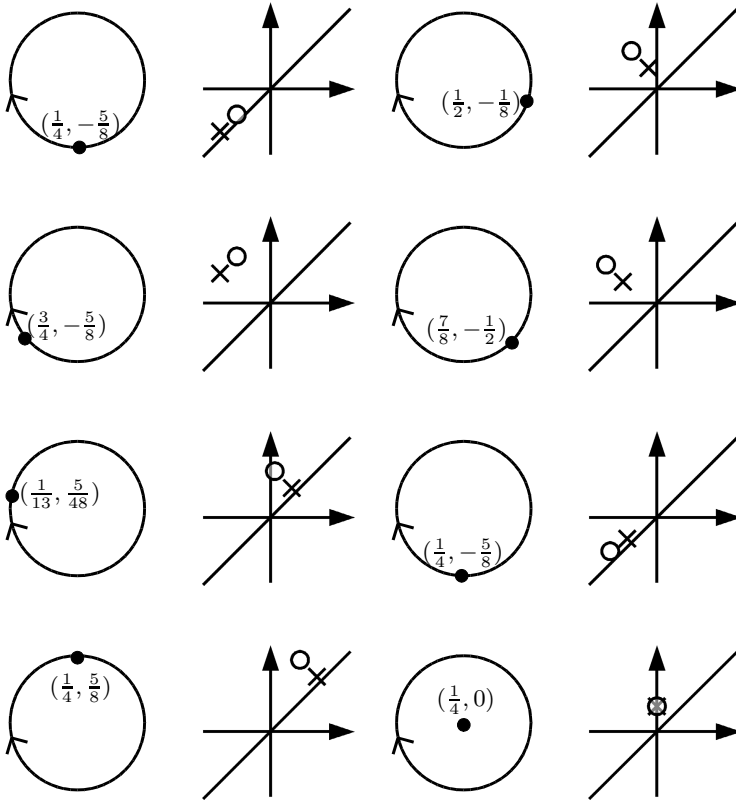


Fig. 4. Codomain of  $\varphi$  for Example 3. Full lines are birth or death points of 0-cycles; dotted lines are  $r_{(m,b)}$  for a few values in  $\text{Adm}_2$ . The loop  $\gamma : I \rightarrow \text{Adm}_2^*(\varphi)$  is also shown.





**Fig. 5.** Schematic of the evolution of the 0th order persistence diagram  $\text{Dgm}(\varphi_\gamma(t))$  for Example 3 as  $t$  goes from 0 to 1. Circle and cross denote the proper cornerpoints. Persistence diagrams are not to scale, but respective positions of cornerpoints are preserved. Similarly, path in  $(a, \beta)$ -space does not actually follow a geometric circle, but is a simple closed curve.

## 4 Conclusions

In this paper we have shown that monodromy can appear in multidimensional topological persistence, and illustrated how it can be managed by specifying the paths to follow in the parameter space  $\text{Adm}_n$  of all admissible pairs. This new phenomenon is expected to reveal itself important in shape comparison, opening the way to the definition of new distances between persistence diagrams based on the idea of matchings continuously dependent on the parameters  $\mathbf{m}, \mathbf{b}$ . The next step in our research will be to study possible methods to construct these new metrics and their application to problems in shape analysis.

## References

1. Bendich, P., Edelsbrunner, H., Kerber, M.: Computing robustness and persistence for images. *IEEE Transactions on Visualization and Computer Graphics* 16(6), 1251–1260 (2010)
2. Biasotti, S., Cerri, A., Frosini, P., Giorgi, D., Landi, C.: Multidimensional size functions for shape comparison. *J. Math. Imaging Vision* 32, 161–179 (2008)
3. Biasotti, S., De Floriani, L., Falcidieno, B., Frosini, P., Giorgi, D., Landi, C., Papaleo, L., Spagnuolo, M.: Describing shapes by geometrical-topological properties of real functions. *ACM Comput. Surv.* 40(4), 1–87 (2008)
4. Bronstein, A., Bronstein, M., Kimmel, R.: *Numerical Geometry of Non-Rigid Shapes*, 1st edn. Springer Publishing Company, Incorporated (2008)
5. Cagliari, F., Di Fabio, B., Ferri, M.: One-dimensional reduction of multidimensional persistent homology. *Proc. Amer. Math. Soc.* 138, 3003–3017 (2010)
6. Carlsson, G.: Topology and data. *Bulletin of the American Mathematical Society* 46(2), 255–308 (2009)
7. Carlsson, G., Zomorodian, A.: The theory of multidimensional persistence. *Discr. Comput. Geom.* 42(1), 71–93 (2009)
8. Cerri, A., Di Fabio, B., Ferri, M., Frosini, P., Landi, C.: Betti numbers in multi-dimensional persistent homology are stable functions. *Math. Method. Appl. Sci.* (in press), doi:10.1002/mma.2704
9. Cerri, A., Ferri, M., Giorgi, D.: Retrieval of trademark images by means of size functions. *Graph. Models* 68(5), 451–471 (2006)
10. Cohen-Steiner, D., Edelsbrunner, H., Harer, J.: Stability of persistence diagrams. *Discr. Comput. Geom.* 37(1), 103–120 (2007)
11. Di Fabio, B., Landi, C.: Persistent homology and partial similarity of shapes. *Pattern Recognition Letters* 33, 1445–1450 (2012)
12. Edelsbrunner, H., Harer, J.: *Computational Topology: An Introduction*. American Mathematical Society (2009)
13. Edelsbrunner, H., Letscher, D., Zomorodian, A.: Topological persistence and simplification. *Discrete Comput. Geom.* 28(4), 511–533 (2002)
14. Edelsbrunner, H., Symonova, O.: The adaptive topology of a digital image. In: 2012 Ninth International Symposium on Voronoi Diagrams in Science and Engineering (ISVD), pp. 41–48 (2012)
15. Frosini, P., Landi, C.: Size functions and formal series. *Appl. Algebra Engrg. Comm. Comput.* 12(4), 327–349 (2001)
16. Paris, S., Durand, F.: A topological approach to hierarchical segmentation using mean shift. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007*, pp. 1–8 (2007)
17. Robins, V., Wood, P.J., Sheppard, A.P.: Theory and algorithms for constructing discrete Morse complexes from grayscale digital images. *IEEE Trans. Pattern Anal. Mach. Intell.* 33(8), 1646–1658 (2011)
18. Zheng, Y., Gu, S., Edelsbrunner, H., Tomasi, C., Benfey, P.: Detailed reconstruction of 3D plant root shape. In: Metaxas, D.N., Quan, L., Sanfeliu, A., Gool, L.J.V. (eds.) *ICCV*. pp. 2026–2033. IEEE (2011)

# Skeleton Extraction of Vertex Sets Lying on Arbitrary Triangulated 3D Meshes

Dimitri Kudelski<sup>1,2,3</sup>, Sophie Viseur<sup>1,3</sup>, and Jean-Luc Mari<sup>1,2</sup>

<sup>1</sup> Aix-Marseille University, France

[jean-luc.mari@univ-amu.fr](mailto:jean-luc.mari@univ-amu.fr)

<http://www.dil.univ-mrs.fr/~mari/>

<sup>2</sup> Information and System Science Laboratory (LSIS), UMR CNRS 7296,

<sup>3</sup> European Center for Research and Teaching in Environmental Geoscience (CEREGE)

**Abstract.** Complex models can be simply described by notions such as skeletons. These robust shape descriptors faithfully characterize the geometry and the topology of an object. Several methods have been developed yet to obtain the skeleton from regular object representations (*e.g.* 2D images or 3D volumes) but only a few attempt to extract the skeleton from unstructured 3D mesh patches. In this article, we extract a skeleton by topological thinning from vertex sets lying on arbitrary triangulated surface meshes in 3D. The key idea comes down to eroding a 2D set located on a discrete 2-manifold. The main difficulty is to transpose the notion of neighborhood from the classical thinning algorithms where the adjacency is constant (*e.g.* 26-adjacency in digital volumes, 8-adjacency in 2D images) to the mesh domain where the neighborhood is variable due to the adjacency of each vertex. Thus we propose a thinning operator dedicated to irregular meshes in order to extract the skeleton of a vertex set. To estimate the robustness of our technique, several tests and an application to the feature line detection are presented as a case-study.

**Keywords:** surface skeleton extraction, topological thinning, irregular mesh.

## 1 Introduction

The skeleton is a robust shape descriptor faithfully characterizing the topology and the geometry of an object. This notion is widely used for various applications such as video tracking [4], shape recognition [12], surface sketching [9], and in many other scientific domains. Several techniques have been proposed to extract the skeleton from binary 2D images [13], 3D closed meshes defining a volume [1], or 3D cubic grids [8]. However few have been dedicated to the extraction of skeletons from a binary information located on an arbitrary triangulated mesh. Rössl *et al.* [10] have presented a method in which some mathematical morphology operators have been ported to triangulated meshes. The main interest of this approach is to combine an efficient computation and

a simple implementation. However, regarding the operator definitions and the underlying algorithm, several drawbacks have been pointed out which mainly lead to unexpectedly disconnected skeletons [7].

### Contributions

In this article, we propose a novel method to extract the skeleton of unstructured mesh patches by a topological thinning process. To figure out the issues of skeletonization of heterogeneous and arbitrary triangulated meshes, we extend the concepts introduced in [10]. The presented approach herein strictly relies on the mesh connectivity to achieve the extraction of the final skeleton. Therefore, for the sake of understanding, the basic method of Rössl *et al.* is described in Section 2 with an assessment of its abilities and drawbacks. Section 3 details the proposed approach and introduces the additional definitions and the novel algorithm. The results of our method including tests on irregular meshes as well as on the performance of the algorithm are shown in Section 4. Finally, an application to the feature line detection is presented in Section 5.

## 2 Basic Notions and Definitions

### 2.1 Position of the Problem

Let  $\mathcal{S}$  be an arbitrary manifold surface represented by an unstructured mesh patch  $\mathcal{M}$  such as  $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ . The sets  $\mathcal{V}$ ,  $\mathcal{E}$ , and  $\mathcal{T}$  correspond, respectively, to the vertices, the edges, and the triangles composing  $\mathcal{M}$ , the piecewise linear approximation of  $\mathcal{S}$ . The vertices are denoted by  $p_i$ , with  $i \in [0; n[$  and  $n = |\mathcal{V}|$  being the total number of vertices of  $\mathcal{M}$ . The neighborhood  $\mathcal{N}$  of a vertex  $p_i$  is then defined as following:

$$\mathcal{N}(p_i) = \{q_j \mid \exists \text{ a pair } (p_i, q_j) \text{ or } (q_j, p_i) \in \mathcal{E}\}. \quad (1)$$

In such a case,  $m_i = |\mathcal{N}(p_i)|$  represents the total number of neighbors of  $p_i$ .

Let now consider a binary attribute  $F$  on each vertex of  $\mathcal{V}$ . The set  $R \subseteq \mathcal{V}$  is then written as follows:

$$\forall p_i \in R \iff F(p_i) = 1. \quad (2)$$

The attribute  $F$  may be defined from beforehand process such as a manual selection, or a thresholding based on geometrical properties (triangle area, principal curvatures, *etc.*). Then, an edge  $e = (p, q)$  belongs to  $R$  if and only if  $p, q \in R$ . Similarly, a triangle  $t = (p, q, r)$  belongs to  $R$  if and only if  $p, q, r \in R$ .

The main objective is to finally develop a technique to extract the skeleton of the set  $R$  by using a topological thinning based on the mesh connectivity.

### 2.2 The Existing Approach

The skeletonization algorithm introduced by Rössl *et al.* consists in an iterative constraint thinning. This relies on a classification of each vertex of  $R$ . The

authors proposed then three vertex types and  $c(p_i)$ , the *complexity* of the vertex  $p_i$  such as:

$$c(p_i) = \sum_{j=0}^{m_i-1} |F(q_j) - F(q_k)|, \tag{3}$$

where  $k = j + 1 \bmod m_i$  and  $q_j, q_k \in \mathcal{N}(p_i)$ .

**Definition 1.** A vertex  $p_i$  is considered as complex if and only if  $c(p_i) \geq 4$ . The set of all complex vertices is named  $C$ .

A complex vertex  $p_i$  thus potentially corresponds to a part of a skeleton branch if  $c(p_i) = 4$ , or a connection through several branches if  $c(p_i) > 4$ .

**Definition 2.** A vertex  $p_i$  is marked as center if and only if  $\mathcal{N}(p_i) \subseteq E$ . The set of all center vertices is named  $E$ .

**Definition 3.** A vertex  $p_i$  is called disk if and only if  $\exists q_j \in \mathcal{N}(p_i), q_j \in E$  that is a center. The set of all disk vertices is named  $D$ .

A *disk* vertex corresponds to a *simple* point: a point that does not modify the expected skeleton topology if it is removed [3]. We denote  $\overline{X}$  the complementary of the set  $X$  in the region  $R$ .

**Definition 4.** The skeleton operator of  $R$  is defined as a constrained thinning:

$$skeletonize(R) = R \setminus (D \cap \overline{C \cup E}). \tag{4}$$

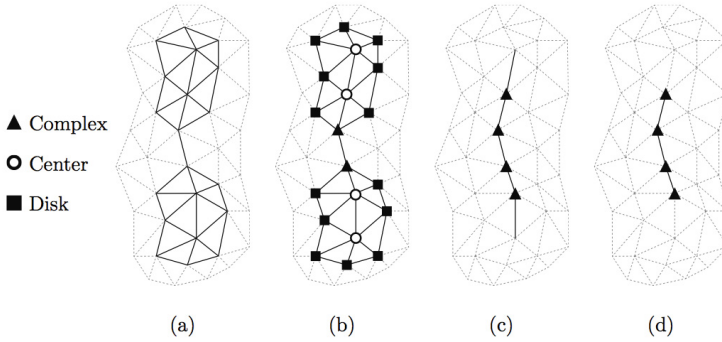
After applying the skeleton operator until idempotence on  $R$ , the set of the remaining vertices, corresponding to the final *skeleton*, is called  $Sk_R$ . During each pass, the skeleton operator removes the boundary *disk* vertices. Figure 1 illustrates the execution of the algorithm. After obtaining the skeleton  $Sk_R$  of  $R$ , it is possible to remove the smallest branches. This last operation is called *pruning* and defined as follows:

$$prune(Sk_R) = Sk_R \setminus \overline{C}. \tag{5}$$

This pruning step is shown by Figure 1 (d).

### 2.3 Result Assessment

Due to the simplicity of the used operators, the computational time of the Rössl *et al.* method is very low, and the skeleton extraction is thus almost instantaneous on meshes composed of 50K triangles. However, the accuracy and the continuity of the obtained skeleton deeply depends on the mesh configuration. In other words, a same set  $R$  defined on two different triangulations of  $S$  could lead to skeletons with two topologies drastically different. Moreover, the lack of continuity also occurs in the case of particular configurations that are shown in



**Fig. 1.** Illustration of the Rössl *et al.* algorithm. From left to right: (a) a set of vertices  $R$ , (b) classification of  $R$ , (c) thinning until idempotence, and (d) resulting skeleton after pruning.

Figure 2 because the removal of *disk* vertices can modify the topology of the skeleton. Figure 3 illustrates the unexpected results and disconnections generated by the execution of the skeletonization. Once the vertices  $P_1$  and  $P_2$  are removed (b), the skeleton becomes disconnected at this location (c). However, some vertices would change to *complex* if a new classification step was applied. This kind of vertices represents relevant points in a topological point of view and thus, should not be deleted.

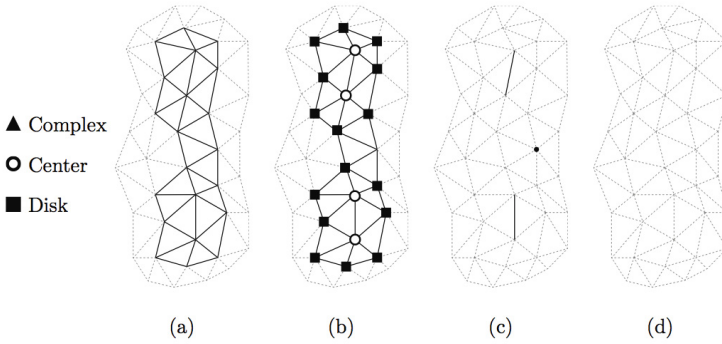
Another issue occurs since pruning is applied: the ending vertices of the skeleton are removed. As a matter of fact, when the set  $R$  contains no *center* and no *complex* vertex, the pruning operator removes all the vertices. This case is illustrated by Figure 4.

### 3 A Skeletonization Method for Any Arbitrary Triangulated Mesh

Both a new definition of particular vertices and a new algorithm have been elaborated to solve the disconnection issues previously raised up in Section 2. These two key points of the approach we propose are successively presented below.

#### 3.1 Additional Definitions

The different classes of vertices proposed by Rössl *et al.* aim at describing the topology of  $R$ . However, they are not sufficient as there are still vertices that are unmarked and that are then not considered in the skeletonization. For this reason, we introduce the *outer* class.



**Fig. 2.** Example of unexpected results by applying the Rössl *et al.* method. From left to right: (a) the set of feature points  $R$ , (b) classification of  $R$ , (c) skeletonization of  $R$ , (d) resulting skeleton after pruning.

**Definition 5.** A vertex  $p_i$  is marked as *outer* if and only if  $F(p_i) = 1$  and  $p_i \notin (C \cup D \cup E)$ . The set of *outer* vertices is named  $O$  and is defined as follows:

$$O = R \setminus (C \cup D \cup E) \tag{6}$$

As it has been shown previously, a vertex may change from one class to another and, as a side-effect, this may lead to potential disconnections during the skeletonization. To counteract this issue, we propose to define a priority between the classes.

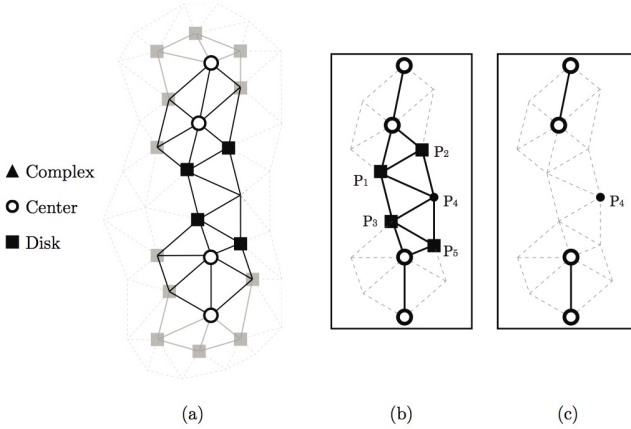
**Definition 6.** The *disk* class has a lower priority over the other classes.

If a vertex is already classified as *disk*, it can change to *complex*, *center* or *outer* if necessary.

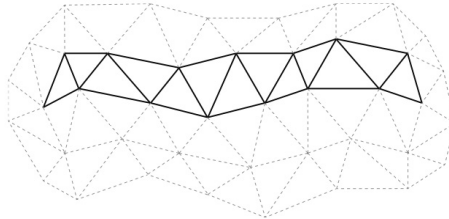
### 3.2 Algorithm

If the skeleton operator defined by Rössl *et al.* is directly applied to an unstructured patch, the final result may suffer from disconnections as some *disk* vertices are deleted while they characterize the topology of the object. To correct this issue, the algorithm we propose does not remove *all* the *disk* vertices but only those that will not be converted to a different priority class after the operator application. This requires to add an additional step in the algorithm: at each application of the skeleton operator, the class of a vertex is recomputed before its deletion. For example, if a *disk* vertex becomes a *complex* vertex, the vertex is not removed.

However, the resulting skeleton may be too thick using this technique (*e.g.* if it is composed of only *outer* vertices). For this reason, a final cleaning step is added to obtain the expected skeleton. At this stage, the skeleton must be composed of *complex* vertices (*i.e.* the skeleton branches or nodes) and *outer*



**Fig. 3.** Execution of the skeletonization operator [10]: (a) vertex classification, (b) execution of the algorithm, (c) final skeleton with a broken topology



**Fig. 4.** Example of a particular configuration: while the vertices of  $R$  are not classified, they will be deleted by the pruning operator of Rössl *et al.*

vertices, the ending points of the branches with only one *complex* vertex in their neighborhood. Thus, to obtain the final skeleton, a two steps process is applied:

- the *outer* vertices that have more than two neighbors belonging to  $R$  are removed;
- the *outer* vertices with at most one neighbor belonging to  $R$  are kept.

Moreover, as for the skeleton operator, each vertex complexity change is checked before removing this vertex. Examples of resulting skeletons are shown in Figure 6 and the impact of the algorithm modification with the update step is presented in Figure 7: *disk* vertices are deleted (b) after checking their classes (c). During the deletion of  $P_1$  and the update step, the class of  $P_2$  changes from *disk* to *complex* and  $P_4$  from *outer* to *complex*. Thus, these vertices are not removed and the extracted skeleton is fully connected and faithfully characterizes the topology of  $R$  (d). The complete method of skeleton extraction is summarized by the algorithm presented on Figure 5.



```

repeat
  forall the vertices  $p_i \in R$  do
    if  $p_i$  is a disk vertex then
      compute the complexity  $c(p_i)$  of the vertex
      if the priority of  $p_i$  does not change then
        delete  $p_i$ 
  until idempotence
repeat
  forall the vertices  $p_i \in R$  do
    if  $p_i$  is an outer vertex then
      compute the complexity  $c(p_i)$  of the vertex
      if the priority of  $p_i$  does not change and if  $|\mathcal{N}(p_i)| > 2$  then
        delete  $p_i$ 
  until idempotence
repeat
  forall the vertices  $p_i \in R$  do
    if  $p_i$  is an outer vertex then
      compute the complexity  $c(p_i)$  of the vertex
      if the priority of  $p_i$  does not change and if  $|\mathcal{N}(p_i)| > 1$  then
        delete  $p_i$ 
  until idempotence

```

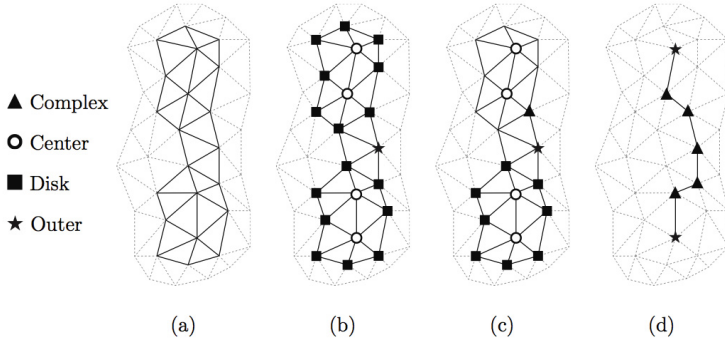
**Fig. 5.** Extraction of the skeleton

## 4 Results

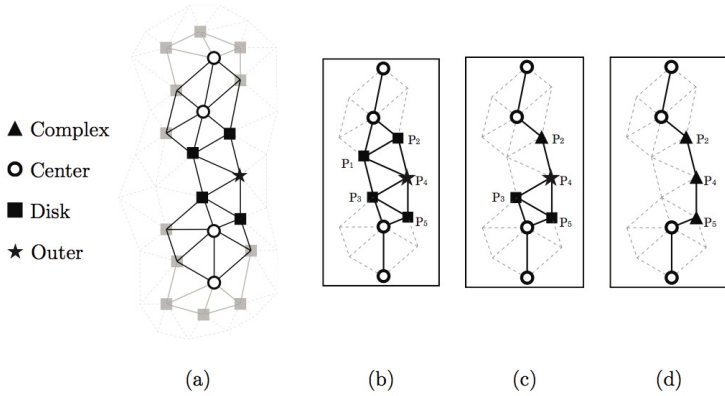
Some results of skeleton extraction on meshes are presented in Figures 8, 9 and 10. The obtained skeletons describe the geometry and the topology of the original set  $R$ . The used meshes are relatively homogeneous in Figure 8 while, in Figures 9 and 10, the algorithm has been tested on irregular meshes to show the robustness of the proposed approach to unstructured meshes. It may be noticed that the resulting skeletons are the expected ones and reflect correctly the topology and geometry of the original set  $R$  in a proper way.

Moreover, since the definitions and the operators used to extract the skeleton are very simple, the computational time of the proposed approach is also very low, even if an additional checking step has been added. It is possible to process a mesh with 100K vertices in 1 second. The tests have been ran on an Intel Core 2 Duo 2.8 Ghz.

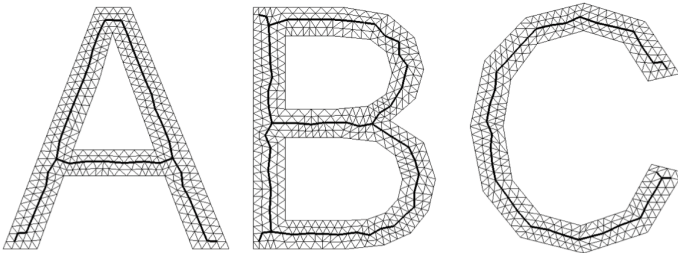
To complete the algorithm tests and to evaluate the robustness of the proposed approach, an application dedicated to the feature line detection is presented in the following section.



**Fig. 6.** Illustration of the proposed approach: (a) region  $R$ , (b) vertex classification, (c) execution of the thinning algorithm with update, (d) final skeleton fully connected



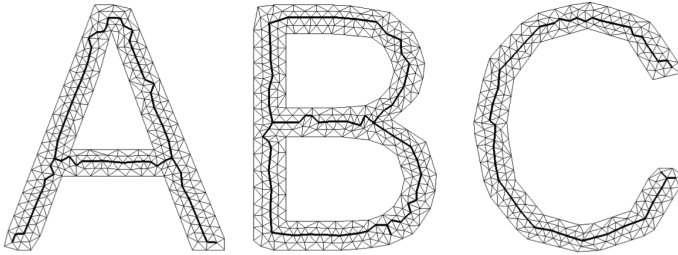
**Fig. 7.** Detailed view of the thinning process: (a) vertex classification, (b) execution of the skeleton operator, (c) update of vertex classes after deletion, (d) final skeleton



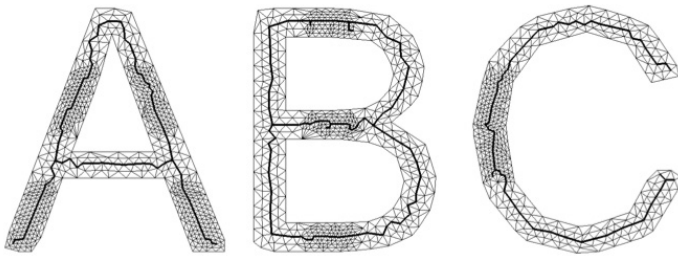
**Fig. 8.** Application of the skeletonization algorithm on regular triangulated 3D meshes

## 5 Application to the Feature Line Detection

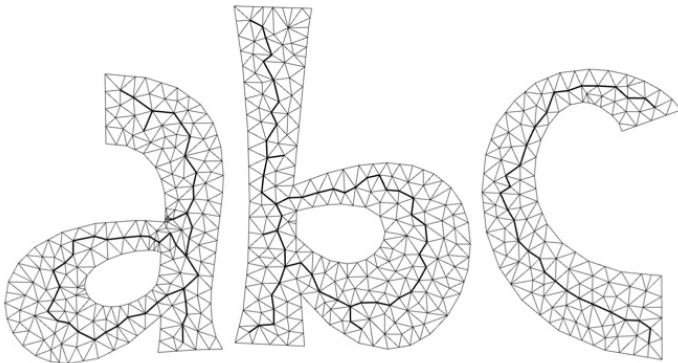
The detection of features within 3D models is a crucial step in shape analysis. It is possible to extract from the surface of an object simple shape descriptors such as lines (drawn on the surface). Generally, the methods of feature line detection focus on the estimation of differential quantities and the research of curvature



**Fig. 9.** Skeleton extraction on irregular 3D meshes

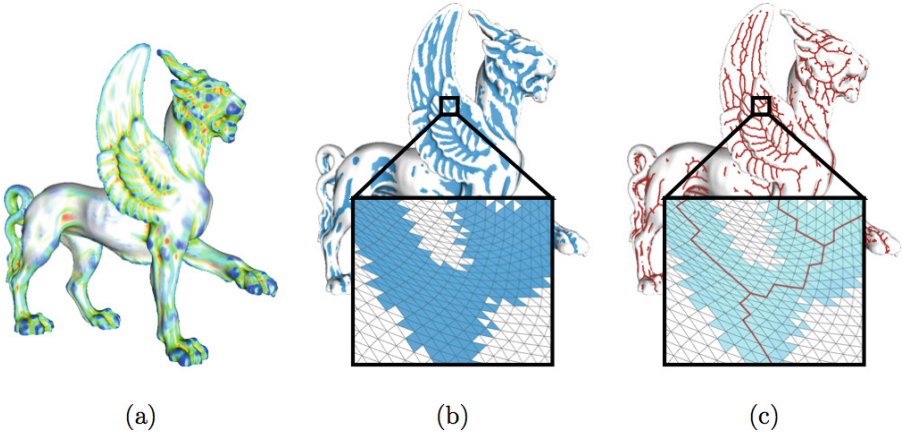


(a)

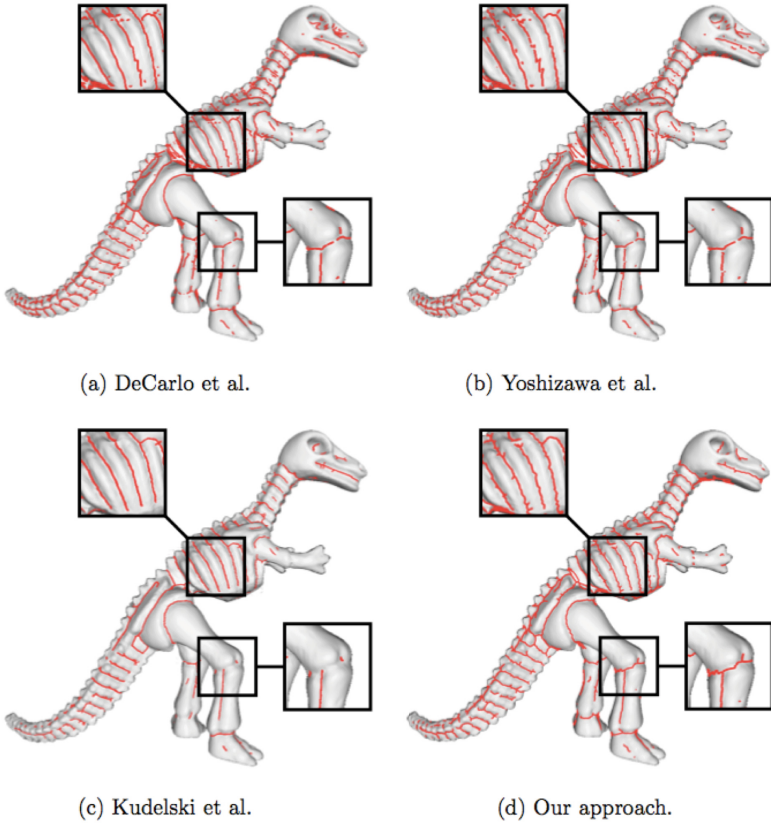


(b)

**Fig. 10.** Extraction of the skeletons on meshes with mixed and unstructured meshes



**Fig. 11.** Algorithm of feature lines extraction: (a) curvature estimation, (b) definition of the set  $R$ , (c) extraction of lines from  $R$  by the proposed thinning approach



**Fig. 12.** Comparison of results obtained from feature detection applied on *Dinosaur*

extrema. However, these techniques are based on *third-order* differential properties and it leads to a common issue: they produce disconnected feature lines because of flat and spherical areas and because of the noise present in data sets. Thus, it is particularly difficult to generate intersections between feature lines. To overcome these recurrent issues, we propose to apply our method to extract salient lines of a model.

In order to define sets over triangulated 3D meshes, we use the algorithm proposed by Kudelski *et al.* [6]. We compute the mean curvature  $H$  through a local polynomial fitting in the least-squares sense [5]. The binary attribute  $F$  is then defined at each vertex  $p_i$  as follows:

$$H_{p_i} > 0 \implies F(p_i) = 1. \quad (7)$$

Finally, the objective is to thin the set, corresponding to potential feature parts of the mesh, in order to obtain lines describing the geometry and the topology of the object.

Figure 11 illustrates the process of feature line detection. The obtained characteristic lines are fully connected and describe accurately the topology of the sets. Then, due to the use of second-order differential properties (*i.e.*, the mean curvatures), the feature extraction is more robust. Moreover, this type of approach allows to generate intersections between feature lines which it is not possible with classical approaches (Figure 12).

## 6 Conclusion

In this article, we have proposed an efficient and general new algorithm to extract in a robust way the skeleton of a set  $R$  defined on a triangulated mesh by topological thinning. This approach relies on the definitions presented by Rössl *et al.* [10]. However, the latter generates, for some mesh configurations, unexpected skeletons that are generally more disconnected than they should. To overcome this issue, an additional definition of vertex categories has been added. Then, we have improved the thinning process by integrating a priority between vertex classes. Tests have been applied on different categories of meshes (homogeneous and heterogeneous) and set configurations. These tests and the application of feature line extraction, presented in the end, illustrate the efficiency of the approach.

As future work, a formal proof based on [2] and issued from the notion of *simple vertices* (by analogy to *simple points*) may need to be considered. Indeed, the Rössl *et al.* article does not include formal validations because the vertices classification is incomplete. With the changes we have made, the *disk* vertices truly correspond to simple points lying on a discrete 2-manifold. Thus it will be possible to transpose the notion of geodesic neighborhood in order to define topological numbers associated with simple vertices.

A second prospect is related to the position of the skeleton nodes. They may be still discussed and further works have to be dedicated to this subject. Indeed, the defined operators do not integrate any geometrical information and

the extraction of the skeleton only relies on a one-ring neighborhood. However, as the position of the skeleton is generally easier to correct than the topology, post-processing steps could be envisaged to optimize the skeleton position. A possible improvement of our method will be to refine the node placement by energy minimization during the extraction to evolve like *active contours*. In this way, the resulting skeleton will describe in a better way both the topology and the geometry of the set lying on the mesh.

**Acknowledgments.** The authors would like to thank Jean Borgomano and Yves Guglielmi of the Geology of Carbonate Systems and Reservoirs laboratory for their precious help and pieces of advice. The models were provided courtesy of Caltech Multi-Res Modeling Group (Feline) and Cyberware (Dinosaur).

## References

1. Au, O.K.C., Tai, C.L., Chu, H.K., Cohen-Or, D., Lee, T.Y.: Skeleton extraction by mesh contraction. *ACM Transaction on Graphics* 27(3), 1–10 (2008)
2. Bertrand, G.: Simple points, topological numbers and geodesic neighborhoods in cubic grids. *Patterns Recognition Letters* 15, 1003–1011 (1994)
3. Bertrand, G.: A boolean characterization of three-dimensional simple points. *Pattern Recognition Letters* 17, 115–124 (1996)
4. Gall, J., Stoll, C., De Aguiar, E., Theobalt, C., Rosenhahn, B., Seidel, H.: Motion capture using joint skeleton tracking and surface estimation. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, pp. 1746–1753. *IEEE Computer Society* (June 2009)
5. Goldfeather, J., Interrante, V.: A novel cubic-order algorithm for approximating principal direction vectors. *ACM Transaction on Graphics* 23(1), 45–63 (2004)
6. Kudelski, D., Mari, J.L., Viseur, S.: 3D feature line detection based on vertex labeling and 2D skeletonization. In: *IEEE International Conference on Shape Modeling and Applications (SMI 2010)*, pp. 246–250. *IEEE Computer Society* (June 2010)
7. Kudelski, D., Mari, J.L., Viseur, S.: Extraction of feature lines with connectivity preservation. In: *Computer Graphics International (CGI 2011 Electronic Proceedings)* (June 2011)
8. Lee, T., Kashyap, R., Chu, C.: Building skeleton models via 3-D medial surface/axis thinning algorithms. *Graphical Models and Image Processing* 56(6), 462–478 (1994)
9. Mari, J.-L.: Surface Sketching with a Voxel-Based Skeleton. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 325–336. *Springer, Heidelberg* (2009)
10. Rössl, C., Kobbelt, L., Seidel, H.P.: Extraction of feature lines on triangulated surfaces using morphological operators. In: *AAAI Spring Symposium on Smart Graphics*, vol. 00-04, pp. 71–75 (March 2000)
11. Siddiqi, K., Pizer, S.: Medial Representations. *Mathematics, Algorithms and Applications. Computational Imaging and Vision*, vol. 37. *Springer* (2008)
12. Yu, K., Wu, J., Zhuang, Y.: Skeleton-Based Recognition of Chinese Calligraphic Character Image. In: Huang, Y.-M.R., Xu, C., Cheng, K.-S., Yang, J.-F.K., Swamy, M.N.S., Li, S., Ding, J.-W. (eds.) *PCM 2008*. LNCS, vol. 5353, pp. 228–237. *Springer, Heidelberg* (2008)
13. Zhang, T.Y., Suen, C.Y.: A fast parallel algorithm for thinning digital patterns. *Communications of the ACM* 27(3), 236–239 (1984)

# Integral Based Curvature Estimators in Digital Geometry<sup>\*</sup>

David Coeurjolly<sup>1</sup>, Jacques-Olivier Lachaud<sup>2</sup>, and Jérémy Levallois<sup>1,2</sup>

<sup>1</sup> Université de Lyon, CNRS  
INSA-Lyon, LIRIS, UMR5205, F-69621, France

<sup>2</sup> Université de Savoie, CNRS  
LAMA, UMR5127, F-73776, France

**Abstract.** In many geometry processing applications, the estimation of differential geometric quantities such as curvature or normal vector field is an essential step. In this paper, we investigate a new class of estimators on digital shape boundaries based on Integral Invariants. More precisely, we provide both proofs of multigrid convergence of curvature estimators and a complete experimental evaluation of their performances.

**Keywords:** Digital geometry, curvature estimation, multigrid convergence, integral invariants.

## 1 Introduction

In many shape processing applications, differential quantities estimation on the shape boundary is usually an important tool. When evaluating a differential estimator on discrete or digital data, we need a way to mathematically link the estimated quantity to the exact Euclidean one. In Digital Geometry, we usually consider multigrid convergence principles: when the shape is digitized on a grid with resolution tending to zero, the estimated quantity should converge to the expected one [4]. Hence, in dimension 2, parameter free convergence results have been obtained for length [3] and normal vector estimation [20]. Based either on binomial convolution principles [15,5], or polynomial fitting [18], convergence results can also be obtained for higher order derivatives of digital curves. Algorithms are parametrized by the size of the convolution or fitting kernel support and convergence theorem holds when such support size is an increasing function of the grid resolution and some shape characteristics. For curvature estimation along 2D curves, multigrid convergence of parameter free estimator is still challenging, although accurate experimental results have been obtained [19]. In 3D, several empirical methods exist for estimating curvatures, but none achieves multigrid convergence (e.g. see [6]).

In geometry processing, interesting mathematical tools have been developed to design differential estimators on smooth surfaces based on integral invariants

---

<sup>\*</sup> This work has been mainly funded by DIGITALSNOW ANR-11-BS02-009 research grants.

[17,16]. They consist in moving a kernel along the shape surface and in computing integrals on the intersection between the shape and the kernel. Authors have demonstrated that some integral quantities provide interesting curvature information when the kernel size tends to zero.

The contributions of the paper can be sketched as follows. First, we define digital versions of integral invariant estimators for which convergence results can be obtained when the grid resolution tends to zero. We provide an explicit formula for the kernel size, which guarantees uniform convergence for smooth enough curves (Sect. 3). Furthermore, we demonstrate that these estimators have efficient implementations and that they compete with classical ones in terms of accuracy (Sect. 4). We also illustrate the strength of the framework to design mean and Gaussian curvature estimators on surfaces in  $\mathbb{Z}^3$ .

## 2 Preliminaries

### 2.1 Shapes, Digital Shapes and Multigrid Convergence

Since we are interested in evaluating both theoretically and experimentally the behavior of a given differential estimator on digital object boundaries, we first have to formalize links between Euclidean objects and digital ones with the help of a digitization process. Let us consider a family  $\mathbb{X}$  of smooth and compact subsets of  $\mathbb{R}^d$ . In Section 3 we will be more precise on the notion of smoothness for shapes  $X \in \mathbb{X}$ . We denote  $D_h(X)$  the digitization of  $X$  in a  $d$ -dimensional grid of resolution  $h$ . More precisely, we consider classical Gauss digitization defined as

$$D_h(X) \stackrel{def}{=} \left( \frac{1}{h} \cdot X \right) \cap \mathbb{Z}^d \quad (1)$$

where  $\frac{1}{h} \cdot X$  is the uniform scaling of  $X$  by factor  $\frac{1}{h}$ . Furthermore, the set  $\partial X$  denotes the frontier of  $X$  (i.e. its topological boundary). If  $z \in \mathbb{Z}^d$ , then  $Q_z$  denotes the unit  $d$ -dimensional cube of  $\mathbb{R}^d$  centered on  $z$ . The  $h$ -frontier  $\Delta_h Z$  of a digital set  $Z \subset \mathbb{Z}^d$  is defined as  $\Delta_h Z \stackrel{def}{=} \partial(h \cdot \cup_{z \in Z} Q_z)$ . Therefore, the  $h$ -frontier of  $D_h(X)$  is a  $d-1$ -dimensional subset of  $\mathbb{R}^d$ , which is close to  $\partial X$ . We will precise the term “close” later in this subsection. Since this paper deals with multigrid convergence, digital shapes will always come from the digitization of continuous shapes. To simplify notations, the  $h$ -frontier of the Gauss digitization at step  $h$  of a shape  $X$  will simply be denoted by  $\partial_h X \stackrel{def}{=} \Delta_h D_h(X)$ , and called later on  $h$ -boundary of  $X$ .

As discussed in various previous works, the idea of multigrid convergence is that when we define a quantity estimator on  $D_h(X)$ , we check if the estimated quantity converges (theoretically and/or experimentally) to the associated one on  $X$  when  $h$  tends to zero. In this paper, we focus on local and global estimated quantities. More formally,

**Definition 1 (Multigrid convergence for local geometric quantities).** *A local discrete geometric estimator  $\hat{E}$  of some geometric quantity  $E$  is multigrid*



convergent for the family  $\mathbb{X}$  if and only if, for any  $X \in \mathbb{X}$ , there exists a grid step  $h_X > 0$  such that the estimate  $\hat{E}(\mathcal{D}_h(X), \hat{x}, h)$  is defined for all  $\hat{x} \in \partial_h X$  with  $0 < h < h_X$ , and for any  $x \in \partial X$ ,

$$\forall \hat{x} \in \partial_h X \text{ with } \|\hat{x} - x\|_\infty \leq h, |\hat{E}(\mathcal{D}_h(X), \hat{x}, h) - E(X, x)| \leq \tau_{X,x}(h), \quad (2)$$

where  $\tau_{X,x} : \mathbb{R}^+ \setminus \{0\} \rightarrow \mathbb{R}^+$  has null limit at 0. This function defines the speed of convergence of  $\hat{E}$  toward  $E$  at point  $x$  of  $X$ . The convergence is uniform for  $X$  when every  $\tau_{X,x}$  is bounded from above by a function  $\tau_X$  independent of  $x \in \partial X$  with null limit at 0.

When a geometrical quantity is global (e.g. area or volume), we do not need explicit mapping between  $\partial X$  and  $\partial_h X$ , and Def. 1 can be rephrased to define *multigrid convergence of global geometric quantities* [4]. A local discrete estimator thus estimates a geometric quantity at points on the  $h$ -frontier of a digital set, otherwise said at any point on the interpixel representation of the digital set boundary. This definition encompasses usual definitions where input points are pointels, linels or surfels. In some proofs, a more precise mapping between points  $x \in \partial X$  and  $\hat{x} \in \partial_h X$  is required. For a 2D shape  $X$  with bounded curvature  $\kappa_{\max}$  along its boundary, this mapping is the *back-projection* map (cf Fig. 1-(right)). Let  $n(X, x, l)$  be the straight segment, centered on  $x$ , aligned with the normal vector at  $x$  along  $\partial X$ , and of half-length  $l$ .

**Definition 2 (Back-projection  $\pi_h^X$  [12]).** For  $0 < h \leq 1/\kappa_{\max}$ , let  $\pi_h^X : \partial_h X \rightarrow \partial X, \hat{x} \mapsto x = \pi_h^X(\hat{x})$ , where  $x$  is the only point such that  $\hat{x} \in n(X, x, \frac{\sqrt{2}}{2}h)$ .

Lemma B.9 [12] indicates that the map  $\pi_h^X$  is well-defined and onto. Lemma B.10 further tells that this map is continuous. It shows that the boundaries  $\partial_h X$  and  $\partial X$  are indeed close, since their Hausdorff distance is no greater than  $\frac{\sqrt{2}}{2}h$ .

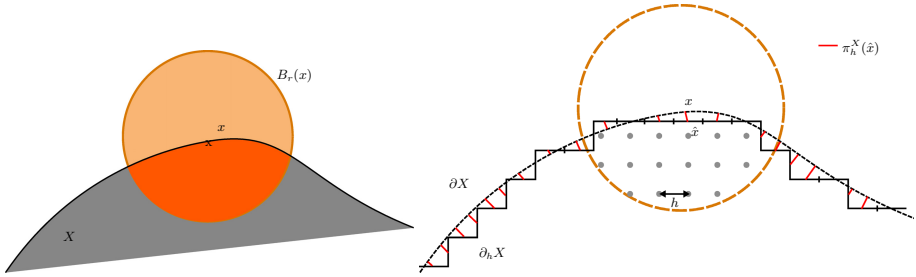
## 2.2 Integral Invariants Theory

In Geometry Processing, integral invariants have been widely investigated to construct estimators of differential quantities (see [17,16] for a complete overview). For short, the main idea is to move a kernel on points  $x \in \partial X$  and to compute integrals on the intersection between  $X$  and the kernel. Even if different kernels (Euclidean ball, Euclidean sphere, ...) and different integration functions can be considered, we focus here on volumetric integral invariants defined as follows:

**Definition 3.** Given  $X \in \mathbb{X}$  and a radius  $r \in \mathbb{R}^{+*}$ , the volumetric integral  $V_r(x)$  at  $x \in \partial X$  is given by (see Fig. 1-(left))

$$V_r(x) \stackrel{def}{=} \int_{B_r(x)} \chi(p) dp, \quad (3)$$

where  $B_r(x)$  is the Euclidean ball with radius  $r$  and center  $x$  and  $\chi(p)$  the characteristic function of  $X$ . In dimension 2, we simply denote  $A_r(x)$  such quantity.



**Fig. 1.** Integral invariant computation (*left*) and notations (*right*) in dimension 2

Several authors have detailed connections between  $V_r(x)$  and curvature (resp. mean curvature) at  $x$  for shapes in  $\mathbb{R}^2$  (resp.  $\mathbb{R}^3$ ) [2,17,16].

**Lemma 1 ([16]).** *For a sufficiently smooth shape  $X$  in  $\mathbb{R}^2$   $x \in \partial X$ , we have*

$$A_r(x) = \frac{\pi}{2}r^2 - \frac{\kappa(X, x)}{3}r^3 + O(r^4) \tag{4}$$

where  $\kappa(X, x)$  is the curvature of  $\partial X$  at  $x$ . For a sufficiently smooth shape  $X$  in  $\mathbb{R}^3$  and  $x \in \partial X$ , we have

$$V_r(x) = \frac{2\pi}{3}r^3 - \frac{\pi H(X, x)}{4}r^4 + O(r^5) \tag{5}$$

where  $H(X, x)$  is the mean curvature of  $\partial X$  at  $x$ .

Such results are obtained by Taylor expansion at  $x$  of the surface  $\partial X$  approximated by a parametric function  $y = f(x)$  in 2D and  $z = f(x, y)$  in 3D. From Eq. (4) and (5) and with a fixed radius  $r$ , one can derive local estimators  $\tilde{\kappa}_r(x)$  and  $\tilde{H}_r(x)$  respectively:

$$\tilde{\kappa}_r(X, x) \stackrel{def}{=} \frac{3\pi}{2r} - \frac{3A_r(x)}{r^3}, \quad \tilde{H}_r(X, x) \stackrel{def}{=} \frac{8}{3r} - \frac{4V_r(x)}{\pi r^4} \tag{6}$$

In this way, when  $r$  tends to zero, both estimated values will converge to expected ones (respectively  $\kappa$  and  $H$ ). More formally:

$$\tilde{\kappa}_r(X, x) = \kappa(X, x) + O(r), \quad \tilde{H}_r(X, x) = H(X, x) + O(r) \tag{7}$$

We mention additional results which allows us to access to directional information such as principal curvature directions. Instead of computing the measure of  $B_r(x) \cap X$  as in Def. 3, we consider its covariance matrix. In [17], authors have demonstrated that eigenvalues and eigenvectors of the covariance matrix provide principal curvature and principal direction information. We do not detail this approach here but we give preliminary results on digital curvature estimators based on this fact in Sect. 4.

When dealing with digital shapes  $D_h(X)$ , implementation of these estimators becomes straightforward: choose a radius  $r$ , center a Euclidean (or digital) ball at chosen points of  $\partial_h X$  (e.g. centroids of linels or surfels), compute the intersection in terms of number of pixels/voxels and finally estimate  $\tilde{\kappa}$  and  $\tilde{H}$  using (6). However, several issues are hidden in this approach: What are meaningful values for  $r$  according to the shape size and geometry ? Do points of  $\partial_h X$  converge to points  $x \in \partial X$  for which Lemma 1 is valid ? Does counting the number of pixels (resp. voxels) converge to  $A_r(x)$  (resp.  $V_r(x)$ ) ? The rest of the paper addresses all these questions.

### 3 Multigrid Convergence of Curvature Estimator in Digital Space

We first recall multigrid convergence theorems for area and volume estimation by counting which will be useful to design digital version of integral invariants. A new digital curvature estimator  $\hat{\kappa}_r$  is then defined (Eq. (11)) and its multigrid convergent properties are established (Theorems 1 and 2).

#### 3.1 Area or Volume Estimation by Counting

Area in the plane and volume in the space can be estimated by counting the number of digital points belonging to the shape. Given digital shapes  $Z \subset \mathbb{Z}^2$  and  $Z' \subset \mathbb{Z}^3$ , the *discrete area and volume estimators by counting at step  $h$*  are defined as  $\widehat{\text{Area}}(Z, h) \stackrel{\text{def}}{=} h^2 \text{Card}(Z)$  and  $\widehat{\text{Vol}}(Z', h) \stackrel{\text{def}}{=} h^3 \text{Card}(Z')$ . Now, if those digital shapes  $Z$  and  $Z'$  come from digitizations of Euclidean shapes  $X$  and  $X'$ , then as the digitization step  $h$  gets finer, these estimators give better and better estimation of the area of  $X$  and of the volume of  $X'$  respectively. We have the following convergence results, letting  $X$  be a finite convex shape of  $\mathbb{R}^2$  and  $X'$  defined similarly in  $\mathbb{R}^3$ :

$$\widehat{\text{Area}}(D_h(X), h) = \text{Area}(X) + O(h^\beta), \quad \widehat{\text{Vol}}(D_h(X'), h) = \text{Vol}(X') + O(h^\gamma), \quad (8)$$

where  $\beta = 1$  in the general case (known since Gauss and Dirichlet according to [10]) and may be improved to  $\frac{15}{11} - \epsilon$ ,  $\epsilon > 0$  arbitrary small, when the shape boundary is  $C^3$  with non-zero curvature [8]. Similar results hold in 3D,  $\gamma = 1$  is the general case (e.g. see [11]) while  $\gamma = \frac{243}{158}$  for smoother boundary [7]. In fact, preceding equations hold whenever the shape boundary can be decomposed in a finite number of convex pieces [9].

#### 3.2 Estimation of Integral Invariants

We are mainly concerned by the estimation of the quantity  $A_r(x) = \text{Area}(B_r(x) \cap X)$  of Def. 3 at a step  $h$ . We cannot readily use Eq. (8) to estimate this area: in this case, the big “O” notation hides the fact that the involved constant depend on the shape size, scale and maximal curvature. It is obvious that doubling the

size of  $X$  will induce a better estimate of the area of  $2 \cdot X$  at the same scale  $h$ . This is a problem with integral invariants, since the involved balls have a radius  $r$  which tends toward 0 as  $h$  tends toward 0. We need to normalize our area estimation so that the error is no more influenced by the scale. Hence we estimate the area  $A_r(x)$  as follows:

$$\begin{aligned} \widehat{\text{Area}}(\mathbb{D}_h(B_r(x) \cap X), h) &\stackrel{\text{def}}{=} h^2 \text{Card}\left(\left(\frac{1}{h} \cdot (B_r(x) \cap X)\right) \cap \mathbb{Z}^2\right), \\ &= h^2 \text{Card}\left(\left(\frac{r}{h} \cdot (B_1\left(\frac{1}{r} \cdot x\right) \cap \frac{1}{r} \cdot X)\right) \cap \mathbb{Z}^2\right), \\ &= r^2 \frac{h^2}{r^2} \text{Card}\left(\left(\frac{r}{h} \cdot (B_1\left(\frac{1}{r} \cdot x\right) \cap \frac{1}{r} \cdot X)\right) \cap \mathbb{Z}^2\right), \\ &= r^2 \widehat{\text{Area}}(\mathbb{D}_{h/r}(B_1\left(\frac{1}{r} \cdot x\right) \cap \frac{1}{r} \cdot X), h/r), \end{aligned}$$

by definitions of  $\widehat{\text{Area}}$  and  $\mathbb{D}$ . We insert (8) in the right handside term:

$$\widehat{\text{Area}}(\mathbb{D}_h(B_r(x) \cap X), h) = r^2 \left( \text{Area}(B_1\left(\frac{1}{r} \cdot x\right) \cap \frac{1}{r} \cdot X) + O((h/r)^\beta) \right). \quad (9)$$

Let  $SB(r)$  denotes the set  $B_1(\frac{1}{r} \cdot x) \cap \frac{1}{r} \cdot X$ . The constant  $K_1$  associated to the big “O” depends only of the maximal curvature of  $\partial SB(r)$ . The curvature is not defined on the subset  $\partial B_1(\frac{1}{r} \cdot x) \cap \frac{1}{r} \cdot \partial X$ , but its influence on the area estimation is negligible (at most  $O(h^2)$ ). The remaining part of  $\partial SB(r)$  has a maximal curvature which is obviously 1 for sufficiently small  $r$ . Indeed, since  $X$  has bounded curvature, its dilated  $\frac{1}{r} \cdot \partial X$  becomes flat at point  $\frac{1}{r} \cdot x$ , the maximal curvature value 1 is thus induced by  $\partial B_1(x)$ . We conclude that there exists some  $r_0$  such that the constant  $K_1$  holds for arbitrary  $r < r_0$ . Developping the big “O” with  $K_1$  and inserting in (9) the straightforward relation  $A_r(x) = \text{Area}(B_r(x) \cap X) = r^2 \text{Area}(B_1(\frac{1}{r} \cdot x) \cap \frac{1}{r} \cdot X)$ , we finally obtain:

$$|\widehat{\text{Area}}(\mathbb{D}_h(B_r(x) \cap X), h) - A_r(x)| \leq K_1 h^\beta r^{2-\beta}. \quad (10)$$

The preceding convergence relation holds for  $h \leq r \leq r_0$ , and is also valid when  $x$  is any point of  $\mathbb{R}^2$ , not necessarily a point of  $\partial X$ . Note that the constant  $K_1$  is independent of the shape  $X$  (but not  $r_0$ ).

The same reasoning is valid in 3D: The curvature is then not defined on the subset  $\partial B_1(\frac{1}{r} \cdot x) \cap \frac{1}{r} \cdot \partial X$ , which tends toward the unit circle as  $r \rightarrow 0$ . The induced error on volume estimation is then the number of intersected voxels ( $\approx 2\pi/h$ ) times the volume of a voxel ( $h^3$ ), and is hence negligible. Maximal curvatures are obviously 1 for sufficiently small  $r$ . The same relation as (10) holds for  $\widehat{\text{Vol}}$ , where  $\beta$  is replaced by  $\gamma$ .

### 3.3 Digital Curvature Estimator

In a similar spirit to (6), we define the *integral digital curvature estimator*  $\hat{\kappa}_r$  of a digital shape  $Z$  at point  $x \in \mathbb{R}^2$  and step  $h$  as:

$$\forall 0 < h < r, \hat{\kappa}_r(Z, x, h) \stackrel{\text{def}}{=} \frac{3\pi}{2r} - \frac{3\widehat{\text{Area}}(B_{r/h}(\frac{1}{h} \cdot x) \cap Z, h)}{r^3}. \quad (11)$$

To establish its multigrid convergence when  $Z$  is the digitization of some subset  $X$  of  $\mathbb{R}^2$ , we proceed in two phases, depending on whether or not we know the exact position of point  $x$  on  $\partial X$  or only an approximation  $\hat{x}$  on  $\partial_h X$ .

**Convergence When  $x \in \partial X$ .** Using relations on integral invariants (6), (10), and relation  $D_h(B_r(x) \cap X) = B_{r/h}(\frac{1}{h} \cdot x) \cap D_h(X)$ , we obtain for  $r < r_0$ :

$$\begin{aligned}
 |\hat{\kappa}_r(D_h(X), x, h) - \kappa(X, x)| &= \left| \frac{3\pi}{2r} - \frac{3\widehat{\text{Area}}(B_{r/h}(\frac{1}{h} \cdot x) \cap D_h(X), h)}{r^3} - \kappa(X, x) \right|, \\
 |\hat{\kappa}_r(D_h(X), x, h) - \kappa(X, x)| &\leq \left| \frac{3\pi}{2r} - \frac{3\text{Area}(B_r(x) \cap X)}{r^3} - \kappa(X, x) \right| + 3K_1 \frac{h^\beta}{r^{1+\beta}} \\
 &\leq |\tilde{\kappa}_r(X, x) - \kappa(X, x)| + 3K_1 \frac{h^\beta}{r^{1+\beta}}, \\
 &\leq O(r) + 3K_1 \frac{h^\beta}{r^{1+\beta}}, \quad (\text{using Eq. (7)}). \tag{12}
 \end{aligned}$$

There are two error terms, both of which depends on the choice of the ball radius  $r$ . We propose to set  $r = kh^\alpha$ , and to choose  $k$  and  $\alpha$  so as to minimize the error bound. Denoting by  $K_2$  the constant in the big “O”, we derive:

$$|\hat{\kappa}_r(D_h(X), x, h) - \kappa(X, x)| \leq K_2 kh^\alpha + \frac{3K_1}{k^{1+\beta}} h^{\beta-\alpha(1+\beta)}. \tag{13}$$

Since one error term in (13) increases with  $\alpha$  while the other decreases with  $\alpha$ , the minimum is achieved when the exponents are the same (solve  $\alpha = \beta - \alpha(1 + \beta)$ ). The constant  $k$  is then obtained by studying its variation at the optimal  $\alpha$ . We obtain the convergence theorem below.

**Theorem 1 (Convergence of digital curvature estimator  $\hat{\kappa}_r$  along  $\partial X$ ).** *Let  $X$  be some convex shape of  $\mathbb{R}^2$ , with at least  $C^2$ -boundary and bounded curvature. Then  $\exists h_0, K_1, K_2$ , such that*

$$\forall h < h_0, r = k_m h^{\alpha_m}, |\hat{\kappa}_r(D_h(X), x, h) - \kappa(X, x)| \leq K h^{\alpha_m}, \tag{14}$$

where  $\alpha_m = \frac{\beta}{2+\beta}, k_m = ((1 + \beta)K_1/K_2)^{\frac{1}{2+\beta}}, K = K_2 k_m + 3K_1/k_m^{1+\beta}$ . When the boundary of  $X$  is  $C^3$  without null curvature points, the exponent  $\alpha_m = \frac{15}{37} - \epsilon \approx 0.405$ , otherwise  $\alpha_m = \frac{1}{3}$ .

**Convergence for  $\hat{x} \in \partial_h X$ .** Unfortunately, the exact position of  $x$  is unknown in digital geometry applications. We only know some digital point  $\hat{x} \in \partial_h X$ , which is close to some point  $x \in \partial X$ . More precisely, the back-projection is used to determine  $x$  as  $\pi_h^X(\hat{x})$ . Integral invariants are not directly applicable since estimator  $\hat{\kappa}_r$  at  $\hat{x}$  is then related to  $A_r(\hat{x})$ , where  $\hat{x}$  does not generally lie onto  $\partial X$ . We have to determine the error between the area measure at  $\hat{x}$  and at  $x$ .

Notice first that point  $\hat{x}$  lies on the normal direction to  $\partial X$  at  $x$ , at a distance  $\delta \stackrel{\text{def}}{=} \|x - \hat{x}\|_2$ . In 3D, we could use Theorem 7 of [17]. In 2D, we achieve similarly:

$$|A_r(\hat{x}) - A_r(x)| = 2r\delta(1 + O(r^2) + O(\delta)). \tag{15}$$

We write (10) at point  $\hat{x}$  (recall that  $A_r(y) = B_r(y) \cap X$ ) and insert (15):

$$\begin{aligned} |\widehat{\text{Area}}(\mathbb{D}_h(B_r(\hat{x}) \cap X), h) - A_r(\hat{x})| &\leq K_1 h^\beta r^{2-\beta}, \quad \text{which implies} \\ |\widehat{\text{Area}}(\mathbb{D}_h(B_r(\hat{x}) \cap X), h) - A_r(x)| &\leq K_1 h^\beta r^{2-\beta} + 2r\delta(1 + O(r^2) + O(\delta)) \end{aligned} \tag{16}$$

In order to get the curvature estimator, we follow the same reasoning as in (12) but we use (16) instead of (10), which gives:

$$|\hat{\kappa}_r(\mathbb{D}_h(X), \hat{x}, h) - \kappa(X, x)| \leq O(r) + 3K_1 \frac{h^\beta}{r^{1+\beta}} + \frac{6\delta}{r^2}(1 + O(r^2) + O(\delta)). \tag{17}$$

We know that  $\delta \leq \frac{\sqrt{2}}{2}h$  (see above). In fact, in some cases (see [13]), we can hope to get a better localization of  $x$  wrt  $\hat{x}$ . Therefore we write  $\delta = O(h^{\alpha'})$ , where  $\alpha' \geq 1$ . We rewrite (17) to obtain an error bound depending only on  $h$  by setting  $r = kh^\alpha$ :

$$\begin{aligned} |\hat{\kappa}_r(\mathbb{D}_h(X), \hat{x}, h) - \kappa(X, x)| &\leq O(h^\alpha) + O(h^{\beta-\alpha(1+\beta)}) \\ &\quad + O(h^{\alpha'-2\alpha}) + O(h^{\alpha'}) + O(h^{2\alpha'-2\alpha}). \end{aligned} \tag{18}$$

We follow the same idea as for (13) to find the best possible parameter  $\alpha$ . The difference is that the optimal  $\alpha_m$  depends not only on  $\beta$  but also on  $\alpha'$ . Simple computations give  $\alpha_m = \frac{\beta}{1+\beta}$  if  $\alpha' \geq \frac{3\beta}{1+\beta}$ , otherwise  $\alpha_m = \frac{\alpha'}{3}$ . If point  $\hat{x}$  is taken on the digital boundary  $\partial_h X$ , then  $\alpha' = 1$  from the relation  $\delta \leq \frac{\sqrt{2}}{2}h$  (see above). We obtain then the convergence theorem below.

**Theorem 2 (Uniform convergence of curvature estimator  $\hat{\kappa}_r$  along  $\partial_h X$ ).** *Let  $X$  be some convex shape of  $\mathbb{R}^2$ , with at least  $C^3$ -boundary and bounded curvature. Then,  $\exists h_0 \in \mathbb{R}^+$ , for any  $h \leq h_0$ , setting  $r = kh^{\frac{1}{3}}$ , we have*

$$\forall x \in \partial X, \forall \hat{x} \in \partial_h X, \|\hat{x} - x\|_\infty \leq h \Rightarrow |\hat{\kappa}_r(\mathbb{D}_h(X), \hat{x}, h) - \kappa(X, x)| \leq Kh^{\frac{1}{3}}.$$

*Proof.* Let  $\hat{x} \in \partial_h X$  and set  $x_0 = \pi_h^X(\hat{x})$ . We know that  $\delta = \|\hat{x} - x_0\|_2 \leq \frac{\sqrt{2}}{2}h$ . Thus  $\alpha' = 1$  and  $\alpha_m = \frac{1}{3}$ . Then (18) becomes:

$$|\hat{\kappa}_r(\mathbb{D}_h(X), \hat{x}, h) - \kappa(X, x_0)| \leq O(h^{\frac{1}{3}}). \tag{19}$$

with  $r = kh^{\frac{1}{3}}$ ,  $k$  is an arbitrary positive constant, and  $r < r_0$  (constant that depends on  $X$ ). This implies  $h < h_1 \stackrel{\text{def}}{=} (r_0/k)^3$ . Let  $x \in \partial X$  with  $\|\hat{x} - x\|_\infty \leq h$ . Since  $\|\hat{x} - x_0\|_2 \leq \frac{\sqrt{2}}{2}h \implies \|\hat{x} - x_0\|_\infty \leq \frac{\sqrt{2}}{2}h < h$ , we conclude that  $x$  and  $x_0$  are under the same closed square  $Q$  of edge length  $2h$  centered on  $\hat{x}$ . It is proven that for sufficiently regular shapes (called  $\text{par}(R)$ -regular shapes in [14],  $R$  is the

inverse of the maximal curvature) there exists a gridstep  $h_2 = \frac{\sqrt{10}}{5}R$  below which the boundary of the shape digitization has same topology as the shape boundary ([12], Theorem B.5). Furthermore, these two boundaries are very close (Hausdorff distance is below  $h$ ). For  $h < R/2 < h_2$ , since  $Q \cap \partial_h X$  is connected,  $Q \cap \partial X$  is connected. Hence  $x$  and  $x_0$  both belongs to the same piece of  $Q \cap \partial X$ , whose length is upper bounded by  $\pi h$ . Since the boundary is  $C^3$  the curvature may only vary between  $x$  and  $x_0$  by some  $O(\pi h)$ . Hence  $|\kappa(X, x) - \kappa(X, x_0)| \leq O(h)$ . Inserting the previous relation in (19) and observing that  $O(h)$  is negligible against  $O(h^{\frac{1}{3}})$  allow us to conclude for  $h < h_0 \stackrel{def}{=} \min(h_1, R/2)$ .  $\square$

**Mean and Gaussian Curvature in 3D.** For the mean curvature, we may follow the same principles as above, using (6) and (7), and inserting the volume estimator  $\widehat{\text{Vol}}$  into the formulas. Since (10) holds for  $\widehat{\text{Vol}}$  (if  $\gamma$  replaces  $\beta$ ), we derive the uniform convergence for the the *integral digital mean curvature estimator*  $\hat{H}_r$  of a digital shape  $Z' \subset \mathbb{Z}^3$  at point  $x \in \mathbb{R}^3$  and step  $h$  as:

$$\forall 0 < h < r, \hat{H}_r(Z', x, h) \stackrel{def}{=} \frac{8}{3r} - \frac{4\widehat{\text{Vol}}(B_{r/h}(\frac{1}{h} \cdot x) \cap Z', h)}{\pi r^4}. \tag{20}$$

If  $X$  has  $C^2$ -boundary and bounded curvatures, the uniform convergence of  $\hat{H}_r$  towards  $H_r$  is achieved along  $\partial X$  for  $r = K'h^{1/3}$ , with a speed of convergence of  $O(h^{1/3})$ . To be valid along  $\partial_h X$ , it is required to prove that the back-projection has the same properties in 3D as in 2D. This is not developed here for space reasons. Similarly, an integral digital Gaussian curvature estimator can be obtained by digital approximation of the covariance matrices of  $X \cap B_r(x)$ . Convergence results rely on the fact that digital moments converge in the same manner as volumes [10].

## 4 Experimental Evaluation

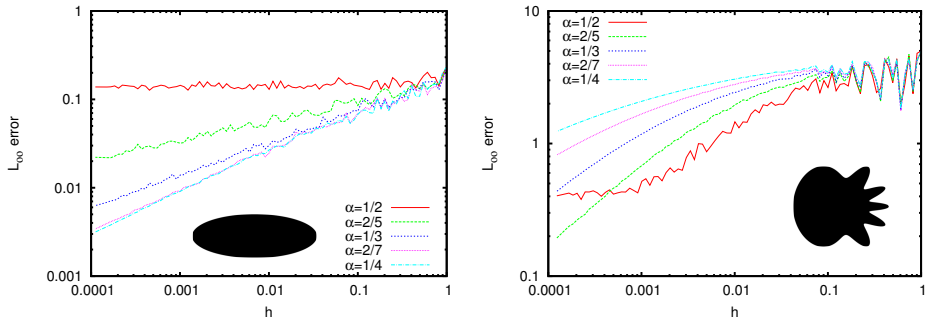
We present an experimental evaluation of curvature estimators in 2D and 3D. We have implemented these Integral Invariant estimators (II) in the `DGtal` library [1] which allows us to have parametric or implicit shape construction in dimension 2 and 3 for multigrid evaluation. Furthermore, it allows comparison with former approaches available in `DGtal`: Most-centered Digital Circular Arc (MDCA) [19] and Binomial based convolution [5].

As described in Sect. 2, brute-force implementation is trivial. We first need to construct a kernel from a Euclidean ball in  $dD$  with radius given by  $r = k_m h^{\alpha_m}$  as described in theorem statements. Then, we track the digital object boundary, center the kernel on each surface elements and compute the volume intersection between the kernel and the object. Using this approach, we obtain a computational cost in  $O((r/h)^d)$  per surface element (*i.e.* the size of the kernel digitization with grid-step  $h$ ). However, we can take benefit from digital surface structure to considerably speed-up this algorithm: if we consider a surface tracker for which surface elements are processed by proximity (the current surface element is a

neighbor of the previous one through a translation vector  $\delta$ ), the area/volume computation can be done incrementally since they are countable additive:

$$\widehat{\text{Area}}(\mathcal{D}_h(X) \cap B_r(x + \delta), h) = \widehat{\text{Area}}(\mathcal{D}_h(X) \cap B_r(x), h) + \widehat{\text{Area}}(\mathcal{D}_h(X) \cap (B_r(x + \delta) \setminus B_r(x)), h) - \widehat{\text{Area}}(\mathcal{D}_h(X) \cap (B_r(x) \setminus B_r(x + \delta)), h).$$

If we precompute all kernels  $\mathcal{D}_h(B_r(0 \pm \delta) \setminus B_r(0))$  for some  $\delta$  displacements (based on surface element umbrella configurations, 8 in 2D and 26 in 3D for  $\|\delta\|_\infty = h$ ), the computational cost per surface element can be reduced to  $O((r/h)^{d-1})$ . Finally, the first surfel has to be computed using kernel  $B_r(\hat{x})$  and the subsequent neighboring surfels are processed using sub-kernels  $\mathcal{D}_h(B_r(0 \pm \delta) \setminus B_r(0))$ .



**Fig. 2.** Comparison of  $h^\alpha$  on an ellipse (left) and on the accelerated flower (right)

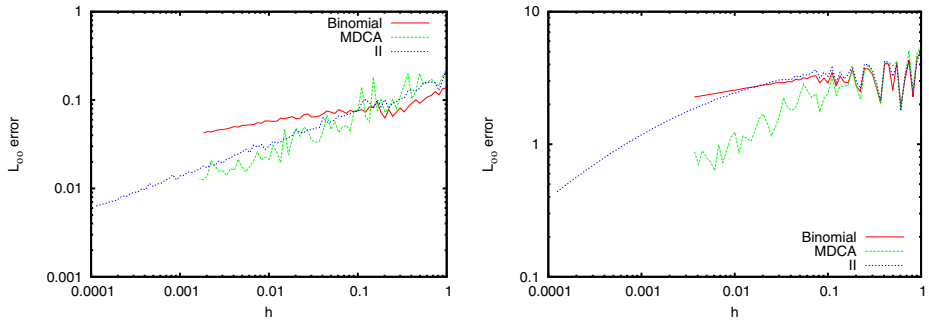
As discussed in the previous section, we first validate the experimental multigrid convergence for various  $\alpha_m$  parameters. Fig. 2 gives results for two 2D shapes: error is given by the  $l_\infty$  distance to the true expected values in order to match with theorem settings. For multigrid ellipses (Fig. 2, left) which corresponds to theorem hypothesis (convex  $C^3$  shape), we observe convergence for several  $\alpha_m$  values. However, as suggested by Theorem 2,  $\alpha_m = \frac{1}{3}$  provides better worst-case errors. Furthermore, note that for  $\alpha_m = \frac{1}{3}$ , the behavior of the  $l_\infty$  error is experimentally in  $O(h^{\frac{1}{3}})$  as suggested by the theorem. For non-convex accelerated flower shape (Fig. 2, right), we still observe the convergence but values  $\alpha_m$  higher than  $\frac{1}{3}$  (and thus larger digital kernel size) seem to lead to lower error values. Further analysis should be done to clearly understand this fact.

In Fig. 3, we compare the proposed 2D curvature estimator (II with  $\alpha_m = \frac{1}{3}$ ) with binomial convolution and MDCA estimator for the  $l_\infty$  error metric. In these noise-free object, MDCA performs better than II or Binomial. However, since II and Binomial are based on integration, we may expect better results on noisy objects. Note that in our experiments, observed convergence speeds on ellipses are:  $O(h^{0.154})$  for binomial,  $O(h^{0.42})$  for MDCA, and  $O(h^{0.38})$  for II using least square linear fitting. The first one differs from theoretical results of [5]. In both graphs, we had to stop the computations for Binomial and MDCA for the following reasons: for our implementation of Binomial, the mask size was too large for small  $h$

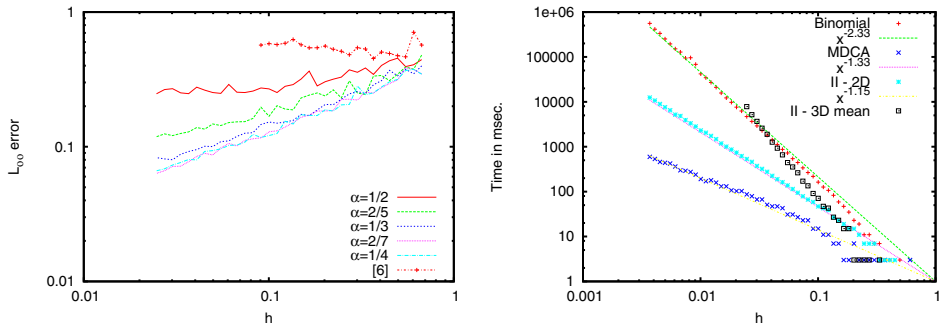


values which induces memory usage issues. For MDCA, circular arc recognition in `DGtal` is driven by a geometrical predicate based on a determinant computation of squared point coordinates. Hence, small  $h$  values lead to numerical capacity issues and thus instability (which could be solved considering arbitrary precision integer numbers but would lead to efficiency issues). The proposed integral invariant estimator does not suffer from these two kind of issues. Fig. 4, right, details timings for the 2D accelerated flower and for the 3D blobby cube (see below). We have performed the same analysis in 3D for the mean curvature: evaluation of  $\alpha_m$  parameters (Fig. 4, left) on a blobby cube<sup>1</sup>. Concerning the literature and as far as we know, no estimators target multigrid convergence. We have compared with fixed neighborhood convolution as described in [6].

Finally, Fig. 5 illustrates mean and Gaussian curvature estimation in 3D (based on covariance matrix of the intersection between the kernel and the shape)

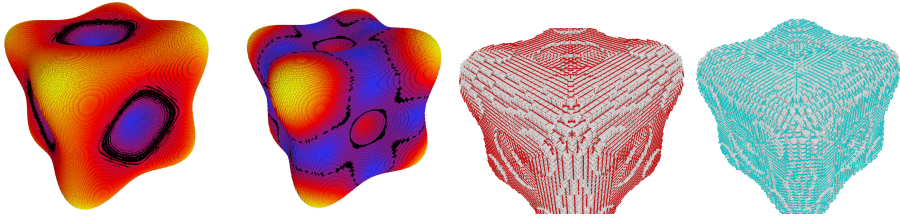


**Fig. 3.** Comparison of  $L_\infty$  error with Binomial [5] and MDCA [19] on multigrid ellipses (left) and accelerated flower (right)



**Fig. 4.** (Left) Experimental evaluation for mean curvature on blobby cube. (Right) Computational efficiency in dimension 2 (blue, cyan and red) and 3 (black).

<sup>1</sup> Implicit surface is  $81x^4 + 81y^4 + 81z^4 - 45x^2 - 45y^2 - 45z^2 - 6 = 0$ .



**Fig. 5.** Illustration of curvature estimation on a 3D blobby cube. *From left to right:* mean curvature and Gaussian curvature mapping ( $h = 0.02$ , highest is yellow, lowest is blue, red is in-between, furthermore we have set to black zero curvature surfels), first and second principal curvature directions.

and principal curvature directions (eigenvectors of the covariance matrix). Concerning mean curvature, setting  $\alpha_m = \frac{1}{3}$  leads to an experimental convergence speed in  $O(h^{0.52})$  for the blobby cube, which means either that  $h^{\frac{1}{3}}$  is not a tight upper bound or that tested parameters  $h$  are not small enough and far from the asymptotic behavior (note that for the finest experiment  $h = 0.0247$ , the object surface contains 1277288 elements).

## 5 Conclusion

In this paper, we have used integral invariant results from differential geometry to design simple and efficient digital curvature estimator in dimension 2 and 3. Digital Geometry is a perfect domain for such differential tools: volume/area computations are digital by nature, interesting connections to fundamental results on Gauss digitization exist, fast computations induced by the specific geometry of digital surfaces. Concerning the 2D curvature estimator, its theoretical convergence speed in  $O(h^{\frac{1}{3}})$  on  $C^3$  contours is comparable to state-of-the-art methods ( $O(h^{\frac{4}{5}})$  for [5] and  $O(h^{\frac{1}{3}})$  for [18]). Evaluation confirms this bound and has demonstrated efficient algorithm in practice with low computational costs. We have also demonstrated that such integral invariants lead to digital mean and Gaussian curvature estimators in 3D. A convergence result for mean curvature has been established and similar results for principal and Gaussian curvatures are expected. Moreover, convergence speed is obtained with a weak constraint on the distance between  $\hat{x}$  and  $x$  (which just needs to be lower than  $h$  for the  $l_\infty$  metric). Using specific projection as discussed in [12], better convergence speed is expected at least for dimension 2.

## References

1. DGtal: Digital geometry tools and algorithms library, <http://libdgtal.org>
2. Bullard, J.W., Garboczi, E.J., Carter, W.C., Fullet, E.R.: Numerical methods for computing interfacial mean curvature. *Computational Materials Science* 4, 103–116 (1995)

3. Coeurjolly, D., Klette, R.: A comparative evaluation of length estimators of digital curves. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 26(2), 252–258 (2004)
4. Coeurjolly, D., Lachaud, J.O., Roussillon, T.: Multigrid convergence of discrete geometric estimators. In: *Digital Geometry Algorithms, Theoretical Foundations and Applications of Computational Imaging*. LNCVB, vol. 2, pp. 395–424. Springer (2012)
5. Esbelin, H.A., Malgouyres, R., Cartade, C.: Convergence of binomial-based derivative estimation for 2 noisy discretized curves. *Theoretical Computer Science* 412(36), 4805–4813 (2011)
6. Fourey, S., Malgouyres, R.: Normals and Curvature Estimation for Digital Surfaces Based on Convolutions. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) *DGCI 2008*. LNCS, vol. 4992, pp. 287–298. Springer, Heidelberg (2008)
7. Guo, J.: On lattice points in large convex bodies. *ArXiv e-prints* (2010)
8. Huxley, M.N.: Exponential sums and lattice points. *Proc. London Math. Soc.* 60, 471–502 (1990)
9. Huxley, M.N.: *Area, lattice points and exponential sums*. Oxford Science Publications (1996)
10. Klette, R., Žunić, J.: Multigrid convergence of calculated features in image analysis. *Journal of Mathematical Imaging and Vision* 13, 173–191 (2000)
11. Krätzel, E.: *Lattice points*, vol. 33. Springer (1988)
12. Lachaud, J.O.: *Espaces non-euclidiens et analyse d’image: modèles déformables riemanniens et discrets, topologie et géométrie discrète*. Habilitation à diriger des recherches, Université Bordeaux 1, Talence, France (2006)
13. Lachaud, J.-O., de Vieilleville, F.: Convex Shapes and Convergence Speed of Discrete Tangent Estimators. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Remagnino, P., Nefian, A., Meenakshisundaram, G., Pascucci, V., Zara, J., Molineros, J., Theisel, H., Malzbender, T. (eds.) *ISVC 2006*. LNCS, vol. 4292, pp. 688–697. Springer, Heidelberg (2006)
14. Latecki, L.J., Conrad, C., Gross, A.: Preserving topology by a digitization process. *Journal of Mathematical Imaging and Vision* 8(2), 131–159 (1998)
15. 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)
16. Pottmann, H., Wallner, J., Huang, Q., Yang, Y.: Integral invariants for robust geometry processing. *Computer Aided Geometric Design* 26(1), 37–60 (2009)
17. Pottmann, H., Wallner, J., Yang, Y., Lai, Y., Hu, S.: Principal curvatures from the integral invariant viewpoint. *Computer Aided Geometric Design* 24(8-9), 428–442 (2007)
18. Provot, L., Gérard, Y.: Estimation of the Derivatives of a Digital Function with a Convergent Bounded Error. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) *DGCI 2011*. LNCS, vol. 6607, pp. 284–295. Springer, Heidelberg (2011)
19. Roussillon, T., Lachaud, J.-O.: Accurate Curvature Estimation along Digital Contours with Maximal Digital Circular Arcs. In: Aggarwal, J.K., Barneva, R.P., Brimkov, V.E., Koroutchev, K.N., Korutcheva, E.R. (eds.) *IWCIA 2011*. LNCS, vol. 6636, pp. 43–55. Springer, Heidelberg (2011)
20. de Vieilleville, F., Lachaud, J.O., Feschet, F.: Maximal digital straight segments and convergence of discrete geometric estimators. *Journal of Mathematical Image and Vision* 27(2), 471–502 (2007)

# A Fast Algorithm for Liver Surgery Planning

Fajie Li<sup>1</sup>, Xinbo Fu<sup>2</sup>, Gisela Klette<sup>3</sup>, and Reinhard Klette<sup>4</sup>

<sup>1</sup> College of Computer Science and Technology  
Huaqiao University, Xiamen, Fujian, China

<sup>2</sup> Xiamen ZhiYe Software Engineering Company Limited, Xiamen, Fujian, China

<sup>3</sup> School of Computing & Mathematical Sciences, Auckland  
University of Technology Private Bag 92006, Auckland 1142, New Zealand

<sup>4</sup> Computer Science Department, The University of Auckland  
Private Bag 92019, Auckland 1142, New Zealand  
li.fajie@hqu.edu.cn

**Abstract.** Assume that a simplified liver model consists of some vein cells and liver cells. Such a liver model contains two kinds of components, the vein component and the liver components, each of them consists of cells which are 26-connected. The vein component has a tree-shape topology. Suppose that the vein component has already been cut into two parts, and one of them is diseased. Liver surgery planning systems need to design an algorithm to decompose the liver components into two kinds of subsets, one (usually just one component) that has been affected by the diseased vein component while the other one is still healthy. So far, existing algorithms depend heavily on surgeons' personal expertise to detect the diseased liver component which needs to be removed. We propose an efficient algorithm for computing the diseased liver component which is based on the diseased vein component, and not on surgeons' personal manipulations.

## 1 Introduction and Related Work

In 2000 it was estimated that liver cancer remains the fifth most common malignancy in men and the eighth in women worldwide, and the number of new cases is 564,000 per year [1]. Liver resection is an often cure for primary liver cancer. The literature reports many liver resection surgical techniques. For example, see [7,10].

Existing liver surgery planning usually requires surgeons' personal expertise to interact during surgery. For example, the planning stage proposed in [10] needs branch labelling which is the most time-consuming step in the planning procedure and usually involves some trial and error on the user's part. *Mint Liver*, a novel 3D image analysis software for liver resection, has to be used by experienced hepatic surgeons for designing the new transection plan. The preoperative planning in [13] calculates the vascular perfusion area using an algorithm based on direction and diameter of the portal vein branch. Reference [8] proposes a probabilistic atlas for liver surgery planning, and [3] discusses a deformable cutting plane for virtual resection where 3D interaction techniques are used to

specify and to modify the clip geometry by medical doctors. The system of [5] relies on the surgeon’s capacity to perform a mental alignment between the resection map and the operating field. The squared Euclidean distance transform was applied in [12] for approximately computing the liver part which should be removed.

In this paper, we apply basic ideas of digital geometry [4] to propose an algorithm for computing the diseased part of a liver. Our algorithm is both time-efficient<sup>1</sup> and “accurate”. The problem to be solved is as follows: Let  $S_l$  be the set of cells in the given 3D input image classified to be liver cells. Set  $S_h$  contains all cells classified to be healthy vein cells. Set  $S_d$  contains all the detected diseased vein cells. We have to calculate that part of the liver which is affected by diseased vein cells.

The *accurate* solution for this problem is defined by the maximum subset  $A \subseteq S_l$  such that  $A$  “is affected” (still to be defined) by the set  $S_d$  of diseased cells. This is an optimization problem. It is solved in this paper by computing exactly three sets  $S_{ab}$ ,  $S_{ah}$ , and  $S_{ad}$  such that  $S_l = S_{ab} \cup S_{ah} \cup S_{ad}$  where  $S_{ab}$  is “affected” by both  $S_h$  and  $S_d$ , and  $S_{a_i}$  is “affected” by  $S_i$  only, for  $i = d, h$ .

Existing algorithms compute only approximately the liver part which should be removed; so far there is no exact specification of the part which should be removed.  $S_{ab}$  can be understood as being a set of boundary cells “between” healthy liver cells and the diseased vein cells.  $S_{ab} \cup S_{ad}$  is finally the set of all liver cells which should be removed.

The paper is structured as follows: In Section 2 we define some notions and notations which are used in our algorithms. In Section 3 we describe and explain the algorithms whose time complexities are analysed in Section 4. We show some experimental results in Section 5 and conclude the paper in Section 6.

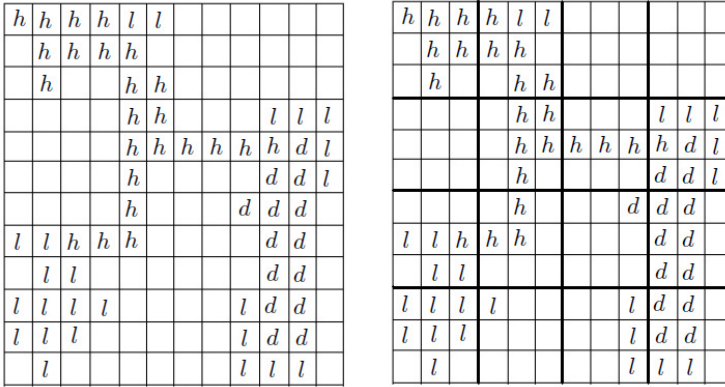
## 2 Basics

Image data are given in a regular 3D grid of grid constant  $\theta_0 > 0$ . We only consider finite sets in this paper. We identify a (grid) *cell* with its centroid, which is a grid point. In this section we discuss the 2D case (i.e. one slice of the 3D data) only; generalization to 3D is straightforward. We also consider a multi-grid approach by varying the given grid constant. Let  $\theta > 0$  be an arbitrary grid constant. Let  $d_e(p, q)$  be the Euclidean distance between two points  $p$  and  $q$  in the plane (e.g. centroids of cells). Given two sets  $A$  and  $B$  in the plane, define  $d_{min}(A, B) = \min\{d_e(p, q) : p \in A \wedge q \in B\}$ . In particular,  $d_{min}(A, B)$  is denoted by  $d_{min}(p, B)$  if  $A$  contains only a single point  $p$ . Define  $d_{max}(A, B) = \max\{d_e(p, q) : p \in A \wedge q \in B\}$ .

We consider healthy vein cells (type- $h$ ), diseased vein cells (type- $d$ ), and liver cells (type- $l$ ), each *cell* being one voxel. Considering the 2D case only in this section, a *cell* is a pixel. Let  $S_i$  be the set of type- $i$  cells, for  $i = d, h, l$ . Let  $S$  be

---

<sup>1</sup> For example, [10] reports about an average labelling time of about 17 minutes, depending on the data set.



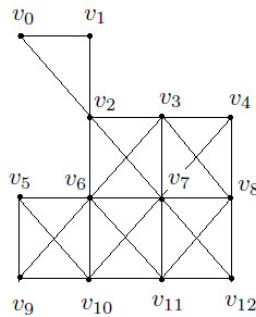
**Fig. 1.** *Left:* A set  $S$  of labelled cells for original grid constant  $\theta_0$ , with  $h$  for type- $h$ ,  $d$  for type- $d$ , and  $l$  for type- $l$ . *Right:* Cells for grid constant  $\theta = \theta_0 \times 3$ .

the union of those three sets. See Fig. 1 on the left. Cells can be uniquely either of *type-d*, *type-h*, *type-l*, or of no assigned type at all (i.e. background cells).

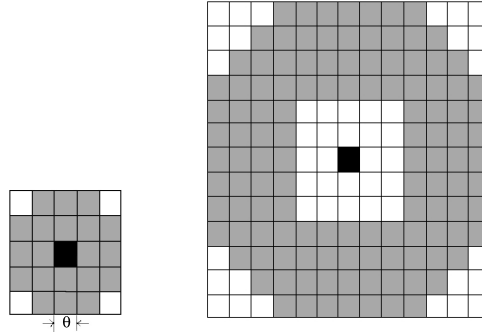
Thus, the three types define sets  $S_d$ ,  $S_h$ , and  $S_l$  of cells in the plane which are pairwise disjoint. We say that set  $S_l$  is *only affected* by  $S_h$  if for each pixel  $p_l \in S_l$ ,  $d_{min}(p_l, S_h) < d_{min}(p_l, S_d)$ ; analogously, we can also have that set  $S_l$  is only affected by  $S_d$ . We say that  $S_l$  is *affected* by both  $S_h$  and  $S_d$  if for each pixel  $p_l \in S_l$ ,  $d_{min}(p_l, S_h) = d_{min}(p_l, S_d)$ .

Let  $\Omega$  be the rectangle of minimum size which contains all the cells of size  $\theta_0 \times \theta_0$  of the given set  $S$ . For a positive integer  $m$ , let  $\theta = \theta_0 \cdot m$ . We analyse  $\Omega$  by using grid constant  $\theta$ . See Fig. 1, right, for an example. Set  $\Omega$  is subdivided into larger  $\theta \times \theta$  cells (*supercells*) which contain several  $\theta_0 \times \theta_0$  cells. Case  $m = 1$  is possible and defines the original constant  $\theta_0$ .

At constant  $\theta$ , set  $S$  can be described by an (undirected weighted)  $\theta$ -graph  $G = [V, E]$  based on 8-adjacency. Each vertex in  $V$  corresponds to one  $\theta \times \theta$  cell which contains at least one of the labelled  $\theta_0 \times \theta_0$  cells of set  $S$ . Two vertices  $v_1$  and  $v_2$  in  $V$  define an edge  $e = \{v_1, v_2\}$  iff the corresponding cells  $C_1$  and  $C_2$



**Fig. 2.** Illustration of the  $\theta$ -graph corresponding to Fig. 1, right. Weights are either  $\theta$  or  $\theta\sqrt{2}$ .



**Fig. 3.** *Left:* adjacency set of a type-*sldoh* supercell  $C$ , containing twenty grey cells. *Right:* adjacency set  $N(C, 3, 5, 2)$  of a type-*sl* supercell  $C$ , shown by grey cells.

are 8-adjacent. Such a graph structure is used in Procedures 1 and 2, and in the main algorithm in Subsection 3.2. See Fig. 2 for an example.

We use standard adjacency definitions of digital geometry to specify four different types of adjacency sets. Consider grid constant  $\theta$ . For a cell  $C$ ,  $L_\infty$ -distances  $i \geq 0$  define layers  $N(C, 1, i)$  of  $\theta \times \theta$  cells around this cell. In general, we have  $8 \times i$  cells in set  $N(C, 1, i)$ , as already discussed in [11]. The four *corner cells* in  $N(C, 1, i)$  have a Euclidean distance  $\sqrt{2} \times \theta \times i$  to cell  $C$ . We call  $N(C, 1, i)$  the *first* adjacency set of supercell  $C$  with radius  $i$  (i.e. cells *near* to  $C$  but not including  $C$  and layers of radius  $j < i$ , thus not a neighbourhood in the sense of topology which would include  $C$ ).

We call  $N(C, 2, i) = \cup_{j=1}^i N(C, 1, j)$  the *second* adjacency set of supercell  $C$  with radius  $i$  (also not including supercell  $C$ ). Furthermore, we also use adjacency sets defined by the Euclidean metric  $L_2$ ; let  $N(C, 3, r)$  be the set of all cells  $C'$  with  $d_{max}(C', C) \leq r$ . We call  $N(C, 3, r)$  the *third* adjacency set of supercell  $C$  with radius  $r$  (not including supercell  $C$ ). So far, this is all very basic digital geometry and just listed here for specifying the used notation.

For our particular application context, we define that an *type-sldoh* cell is one supercell (i.e. with edge length  $\theta_0 \cdot m$ ) that contains at least one type-*l* cell (with edge length  $\theta_0$ ) but also at least one type-*h* or type-*d* cell. A *type-sl* cell is one supercell that contains type-*l* cells only. For example, the  $\theta \times \theta$  cell in Fig. 1 (right), corresponding to vertex  $v_1$  in Fig. 2, is of type-*sldoh* because it contains five type-*h* cells and two type-*l* cells; the cell in Fig. 1 (right), corresponding to  $v_9$ , is of type-*sl* because it contains seven cells which are all of type-*l*. The following adjacency set definitions for type-*sldoh* or type-*sl* cells are motivated by the particular application, and they have been heuristically derived from the given (extensive) image data. Those adjacency sets can be modified without affecting the basic ideas of the algorithms. Assume grid constant  $\theta$ . The *adjacency set* of a type-*sldoh* supercell  $C$  is defined by  $N(C, 3, 2 \times \sqrt{2} \times \theta)$ . See Fig. 3, left. We make use of this in Lines 1–8 of Procedure 1 as shown in Fig. 6. For defining the

adjacency set of a type- $sl$  supercell  $C$ , let  $m$  and  $n$  be two non-negative integers with  $n < m$ . Set

$$N(C, 4, m, n) = N(C, 3, m(\sqrt{2} \times \theta)) \setminus N(C, 2, n)$$

is the *fourth* adjacency set of supercell  $C$ . Figure 3, right, illustrates a set  $N(C, 4, 5, 2)$ . We make use of this in Lines 1–9 of Procedure 2.

Definitions given in this section can be generalized for the 3D case, and we do not specify them here because those generalizations are straightforward.

### 3 Algorithms

Assume that we have a number of slices of 2D images from a  $CT$ -scan. Each 2D image contains  $\theta_0 \times \theta_0$  cells of type- $d$ , type- $h$ , type- $l$ , or “other” cells (that is, background cells). Each type- $h$  cell represents a healthy vein cell. Each type- $d$  cell represents a diseased vein cell. Each type- $l$  cell represents a liver cell. Our goal is to classify type- $l$  cells  $p_l \in S_l$  depending on the value  $d_{min}(p_l, S_h)$  and  $d_{min}(p_l, S_d)$ .

In this section, we describe a naive brute-force algorithm (Algorithm 1), its improved version (Algorithm 2), and then a more efficient main algorithm (Algorithm 3). These three algorithms are used to classify type- $l$  cells based on type- $d$  and type- $h$  cells. As usual,  $S_i$  is the set of type- $i$  cells, for  $i = d, h, l$ .

#### 3.1 A Brute-Force Algorithm and Its Improved Version

The idea of Algorithm 1 is simple. We scan through the set  $S_l$  of all type- $l$  cells. For each cell  $p_l \in S_l$ , we decide to which subset ( $S_{a_b}$ ,  $S_{a_d}$  or  $S_{a_h}$ ) cell  $p_l$  belongs to by simply testing the values of  $d_{min}(p_l, S_i)$ , for  $i = d, h$ : If  $d_{min}(p_l, S_h) < d_{min}(p_l, S_d)$ , then let  $p_l$  be in  $S_{a_h}$ ; else, if  $d_{min}(p_l, S_h) > d_{min}(p_l, S_d)$ , then let  $p_l$  be in  $S_{a_d}$ ; otherwise let  $p_l$  be in  $S_{a_b}$ . The pseudocode is given in Fig. 4.

**Algorithm 1.** (A brute-force algorithm for separating type- $l$  cells)

*Input:* Three sets  $S_d$ ,  $S_h$ , and  $S_l$  such that  $S_i$  contains type- $i$  cells, where  $i = d, h, l$ .

*Output:* Three sets  $S_{a_b}$ ,  $S_{a_d}$  and  $S_{a_h}$  such that  $S_l = S_{a_b} \cup S_{a_d} \cup S_{a_h}$ , where  $S_{a_b}$  is affected by both  $S_d$  and  $S_h$ , and  $S_{a_i}$  is affected by  $S_i$  only, for  $i = d, h$ .

*Pseudocode:* See Fig. 4.

The idea of Algorithm 2 is also simple. We may not have to go through each grid point in  $S_d$ . If there exists a cell  $p_d$  such that  $d_e(p_l, p_d) < d_{min}(p_l, S_h)$  then let  $p_l$  be in  $S_{a_d}$ , and break both this for-loop and the outer for-loop, and test the next cell after  $p_l$  in  $S_l$ . The pseudocode of Algorithm 2 is modified from the code of Algorithm 1 by inserting a few lines after Line 3 in Fig. 4. It is described in Fig. 5.

**Algorithm 2.** (An improved version of Algorithm 1)

Input and output are the same as for Algorithm 1 but for set cardinalities assume that  $|S_h| \leq |S_d|$ .

*Pseudocode:* See Fig. 5.



```

1: Let  $S_{a_b} = S_{a_d} = S_{a_h} = \emptyset$ .
2: for each  $p_l \in S_l$  do
3:   Go through  $S_h$  for computing  $d_{min}(p_l, S_h)$ .
4:   Go through  $S_d$  for computing  $d_{min}(p_l, S_d)$ .
5:   if  $d_{min}(p_l, S_d) = d_{min}(p_l, S_h)$  then
6:      $S_{a_b} = S_{a_b} \cup \{p_l\}$ 
7:   else
8:     if  $d_{min}(p_l, S_d) < d_{min}(p_l, S_h)$  then
9:        $S_{a_d} = S_{a_d} \cup \{p_l\}$ 
10:    else
11:       $S_{a_h} = S_{a_h} \cup \{p_l\}$ 
12:    end if
13:  end if
14: end for
15: Return  $S_{a_b}$ ,  $S_{a_d}$ , and  $S_{a_h}$ .

```

**Fig. 4.** A brute-force algorithm for separating type- $l$  cells (pseudocode of Algorithm 1)

```

1: Lines 1–3 from Fig. 4.
2: for each  $p_d \in S_d$  do
3:   if  $d_c(p_l, p_d) < d_{min}(p_l, S_h)$  then
4:      $S_{a_d} = S_{a_d} \cup \{p_l\}$ 
5:     Break both this for-loop and the outer for-loop.
6:   end if
7: end for
8: Exactly copy Lines 4–15 from Fig. 4, but remove Line 7.

```

**Fig. 5.** An improved version of Algorithm 1 for separating type- $l$  cells: Simply insert Lines 3–6 in the code of Algorithm 2 after Line 3 in the code of Algorithm 1. Note that the ‘outer for-loop’ refers to the outer loop as specified in Algorithm 1.

### 3.2 Algorithm in 2D

Algorithm 3, our main algorithm, is based on Algorithm 2 and Procedures 1 and 2. Procedure 1 is used to compute relevant adjacent cells within the adjacency set of a type-*sldoh* supercell. The procedure is shown in Fig. 6. The word *relevant* means here that each returned supercell is both in the adjacency set of the given type-*sldoh* supercell as well as of the corresponding supercell of the  $\theta$ -graph, for grid constant  $\theta \geq \theta_0$ .

**Procedure 1.** (Compute relevant supercells in adjacency set of a type-*sldoh* supercell)

*Input:* A  $\theta$ -graph  $G = [V, E]$ , and a type-*sldoh* supercell represented by vertex  $v \in V$ .

*Output:* Return a subset  $N_v$  of  $V$  such that  $d_{min}(v', v) \leq \theta$ , for each supercell  $v' \in N_v$ .

*Pseudocode:* See Fig. 6.

```

1: Let  $N_i$  be the sets of supercells of the first two adjacency sets of supercell  $v$ , where
    $i = 1, 2$ .
2: Let  $N'_2 = \emptyset$ .
3: for each supercell  $u \in N_2$  do
4:   if  $d_{min}(u, v) \leq \theta$  then
5:      $N'_2 = N'_2 \cup \{u\}$ 
6:   end if
7: end for
8: Let  $N = N_1 \cup N'_2$ .
9: Let  $N_v = \emptyset$ .
10: for each supercell  $u \in N$  do
11:   if  $u \in V$  then
12:      $N_v = N_v \cup \{u\}$ 
13:   end if
14: end for
15: Return  $N_v$ .

```

**Fig. 6.** Computation of relevant cells adjacent to a type-*sldoh* supercell  $v$  (Procedure 1)

The following Procedure 2 is used to compute relevant supercells in the adjacency set of a type-*sl* supercell. The word *relevant* means here that the returned supercells are both in the adjacency set of the type-*sl* supercell and the supercells of the  $\theta$ -graph. The pseudocode is shown in Fig. 7.

**Procedure 2.** (Compute relevant supercells in the adjacency set of a type-*sl* supercell)

*Input:* A  $\theta$ -graph  $G = [V, E]$ , a type-*sl* supercell  $v \in G$ . Assume that there exist type-*d* or type-*h* cells.

*Output:* Return a subset  $N_v$  of  $V$  such that, for each supercell  $v' \in N_v$ , we have that  $d_e(v', v) \leq R_v$ , where  $R_v$  is the radius of  $N(v, 3, R_v)$ .

*Pseudocode:* See Fig. 7.

The imaged part of the liver is defined by all type-*l* cells. Its veins consists of type-*i* cells, where  $i = d, h$ .

The main idea of the following main algorithm (Algorithm 3) is to decompose the liver into some supercells so as to reuse the improved version of the above brute-force algorithm (i.e., Algorithm 2) “locally” by removing unnecessary type-*i* cells (where  $i = d, h$ ) which are “too far” from the current supercell (thus, also “too far” from any type-*l* cells contained in the current supercell).

**Algorithm 3.** (Main Algorithm)

*Input:* A set  $S$  containing type-*i* cells, where  $i = d, h, l$ , and a parameter  $m > 0$  (for example,  $m = 20$ ).

*Output:* Three sets  $S_{ab}$ ,  $S_{ad}$ , and  $S_{ah}$  such that  $S_l = S_{ab} \cup S_{ad} \cup S_{ah}$ , where  $S_{ab}$  is affected by both  $S_d$  and  $S_h$ , and  $S_{3_i}$  is affected by  $S_i$  only, for  $i = d, h$ .

*Pseudocode:* See Fig. 8.

Regarding a proof of the correctness of Algorithm 3, at first, the set of liver cells (i.e. of their centroids) can be assumed to be digitally convex (i.e. the Gauss

```

1:  $i = 1$ 
2: while there is not any type- $d$  or type- $h$  cell contained in a supercell in  $N(v, 1, i)$ 
   (i.e., the first adjacency set with distance  $i$  of the supercell  $v$ ) do
3:    $i = i + 1$ 
4: end while
5: Take any corner supercell  $u$  in  $N(v, 1, i)$ .
6: Let  $R_v = d_{max}(u, v)$ .
7: Compute  $N(v, 3, R_v)$  (i.e., the third adjacency set of the supercell  $v$  with radius
    $R_v$ ).
8: Compute  $N(v, 2, i)$  (i.e., the second adjacency set of the supercell  $v$  with radius  $i$ ).
9: Let  $N_4 = N(v, 3, R_v) \setminus N(v, 2, i)$ .
10: Let  $N_v = \emptyset$ .
11: for each supercell  $u \in N_4$  do
12:   if  $u \in V$  then
13:      $N_v = N_v \cup \{u\}$ 
14:   end if
15: end for
16: Return  $N_v$ .

```

**Fig. 7.** Computation of relevant supercells adjacent to a type- $sl$  supercell  $v$  (Procedure 2)

digitization of a convex polyhedron). Thus we can define *affected* by using  $d_{min}$  as in Section 2, based on the Euclidean distance  $d_e$ .

For each supercell  $C$ , if  $C$  is of type- $sldoh$  supercell then, for any two original (i.e., before digitization in Line 3) cells  $p_1$  and  $p_2$  contained in  $C$ , for their distance we have that  $d_e(p_1, p_2) < \sqrt{2} \cdot \theta_0$ . Thus, we can only consider type- $i$  cells inside of  $N(C, 3, \sqrt{2} \cdot \theta)$  for separating type- $l$  cells in  $C$ , for  $i = d, h$ . In short, the candidate sets are reduced from  $S_d$  and  $S_h$  to  $S_d \cap N(C, 3, \sqrt{2} \cdot \theta)$  and  $S_h \cap N(C, 3, \sqrt{2} \cdot \theta)$ .

For each supercell  $C$ , if  $C$  is of type- $sl$  then for any two original cells  $p_1$  and  $p_2$  contained in  $C$  and the corner supercell  $C'$  (See Fig. 9 for an illustration of  $C$  and  $C'$ ), we have that  $d_{max}(p_1, p_2) \leq R_C$ . Note that the radius  $R_C = d_{max}(C', C)$  is defined in Line 6 in Procedure 2. Thus, we can only consider type- $i$  cells inside of  $N_C$  for separating the type- $l$  cells in  $C$ , where  $i = d, h$  ( $N_C$  is the subset returned by Procedure 2).

In short, the candidate sets are reduced from  $S_1$  and  $S_2$  to  $S_1 \cap N(C, 3, \sqrt{2} \times \theta)$  and  $S_2 \cap N(C, 3, \sqrt{2} \times \theta)$ , respectively. See Fig. 9 for an illustration of  $R_C$  and  $N_C$ .

### 3.3 Algorithm in 3D

The limited space does not allow a full description. However, the algorithms are very straightforward extensions of the 2D case: copy from Subsection 3.2 and replace  $\sqrt{2}$  by  $\sqrt{3}$ . Figure 10 shows some experimental results of our main algorithm in 3D.

```

1: Let  $\theta = m \times \theta_0$ .
2: Let  $\Omega$  be the smallest isothetic circumscribing rectangle that contains all cells of
   set  $S$ .
3: Digitize  $\Omega$  using grid constant  $\theta$  and label a supercell  $C$  to be “active” if  $C$  contains
   type- $i$  cells, where  $i = d, h, l$ .
4: Construct the  $h$ -graph  $G = [V, E]$  as follows: Let  $V$  be all “active” supercells. For
   any two supercells  $C_1$  and  $C_2$  in  $V$ , define an edge  $e = \{C_1, C_2\}$  if  $C_1$  and  $C_2$  are
   8-adjacent.
5: Let  $S_{a_b} = S_{a_d} = S_{a_h} = \emptyset$ .
6: for each supercell  $v \in V$  do
7:   if  $v$  is a type-sldoh supercell then
8:     Let  $G$  and  $v$  be the input for Procedure 1 for computing relevant adjacent
       supercells  $N_v$  for the type-sldoh supercell  $v$ .
9:     Let  $S_d = S_h = \emptyset$ .
10:    Let  $S_l$  be the set of all cells in  $v$ .
11:    for each supercell  $u \in N_v \cup \{v\}$  do
12:      for each cell  $p \in u$  do
13:        if  $p$  is type- $d$  cell then
14:           $S_d = S_d \cup \{p\}$ 
15:        else
16:          if  $p$  is type- $h$  cell then
17:             $S_h = S_h \cup \{p\}$ 
18:          end if
19:        end if
20:      end for
21:    end for
22:    Let  $S_d, S_h$  and  $S_l$  be the input for Algorithm 2 for computing three sets
        $S_{a_b}(v), S_{a_d}(v)$  and  $S_{a_l}(v)$  such that  $S_l = S_{a_b}(v) \cup S_{a_d}(v) \cup S_{a_l}(v)$ , where
        $S_{a_b}(v)$  is affected by both  $S_d$  and  $S_h$ , and  $S_{3_i}(v)$  is affected by  $S_i$  only, for
        $i = d, h$ .
23:     $S_{3_i} = S_{3_i} \cup S_{3_i}(v)$ , where  $i = d, h, l$ .
24:  else
25:    if  $v$  is a type-sl supercell then
26:      Let  $G$  and  $v$  be the input for Procedure 2 for computing relevant adjacent
        supercells  $N_v$  for the type-sl supercell  $v$ .
27:      Exactly copy Lines 9–23 into here.
28:    end if
29:  end if
30: end for
31: Return the three sets  $S_{a_b}, S_{a_d}$ , and  $S_{a_l}$ .

```

**Fig. 8.** Pseudocode of the main algorithm:  $m$  is a parameter and can be adjusted depending on the size and distribution of the input data set  $S$

## 4 Time Complexity

Regarding the time complexity of Algorithms 1 and 2, and of Procedures 1 and 2, we have the following:

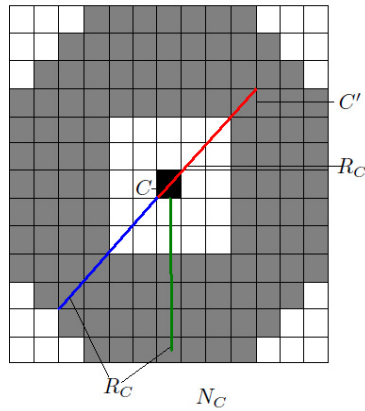


Fig. 9. Illustration for the correctness proof of Algorithm 3

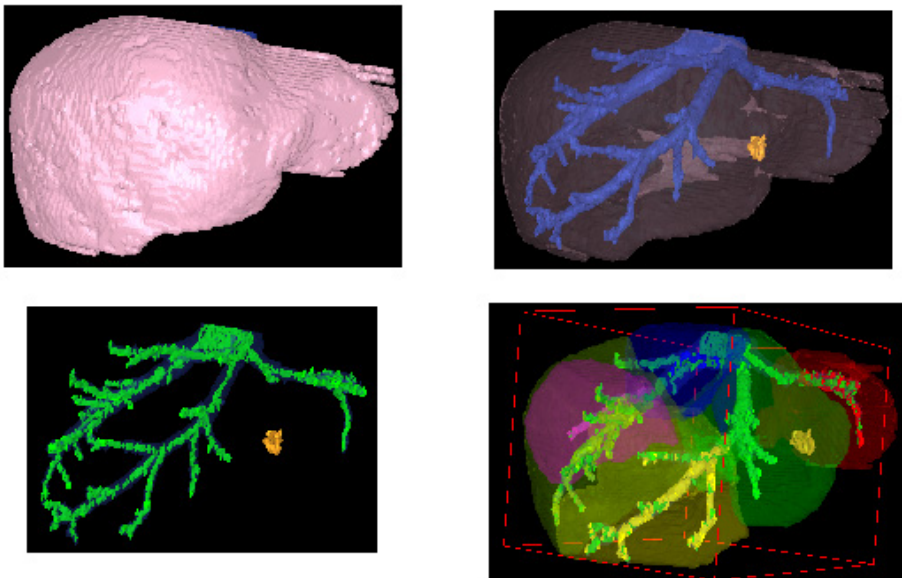


Fig. 10. *Left, top:* The liver model. *Right, top:* The vein component which has a tree shape topology, and the tumour. *Left, bottom:* The cells in the vein component and the tumour. *Right, bottom:* A liver usually consists of eight parts shown in eight colours. Only the red part on the right is the diseased part, as detected by our algorithm, and it should be removed. It seems there are some green cells between red cells. This may disappear if we change the angle of view.

**Lemma 1.** *Algorithm 1 takes  $|S_3| \cdot (|S_1| + |S_2|)$  operations.*

**Lemma 2.** *Algorithm 2 takes at most  $|S_3| \cdot (|S_1| + |S_2|)$  operations.*

**Table 1.** The results of five experiments are shown from left to right, organized in columns. Parameter  $m$  is applied in Line 1 of Algorithm 3, and the time is in seconds for a 3D voxel data set of dimensions  $324 \times 243 \times 129$ .

$m$	Time	$m$	Time	$m$	Time	$m$	Time	$m$	Time
0	53	0	167	0	69	0	53	0	49
10	24	10	36	10	12	10	36	10	33
20	24	20	23	20	14	20	37	20	32
40	38	40	41	40	23	40	57	40	50

**Lemma 3.** *Procedure 1 takes  $\mathcal{O}(|N|)$  operations, where  $N$  is defined as in Line 8 of Procedure 1.*

It is  $N = N(v, 3, \sqrt{2} \times \theta)$  in the 2D case, and  $N = N(v, 3, \sqrt{3} \times \theta)$  in the 3D case. For any integer parameter  $m \geq 1$  it is  $|N| = 20$  in the 2D case, and  $|N| = 80$  in the 3D case.

**Lemma 4.** *Procedure 2 takes  $\mathcal{O}(|N(v, 3, R_v)|)$  operations, where  $R_v$  is defined in Line 6 of Procedure 2.*

Regarding the time complexity of the main algorithm (Algorithm 3), the main computations occur in Lines 8, 22, and 26.

By Lemma 3, the computation in Line 8 takes  $\mathcal{O}(n \times |N|)$  operations, where  $n$  is the number of supercells.

By Lemma 4, the computation in Line 26 takes  $\mathcal{O}(n \times n_{max})$  operations,  $n_{max}$  is the maximal value of all  $|N(v, 3, M_{R_v})|$ 's, and  $R_v$  is defined in Line 6 in Procedure 2.

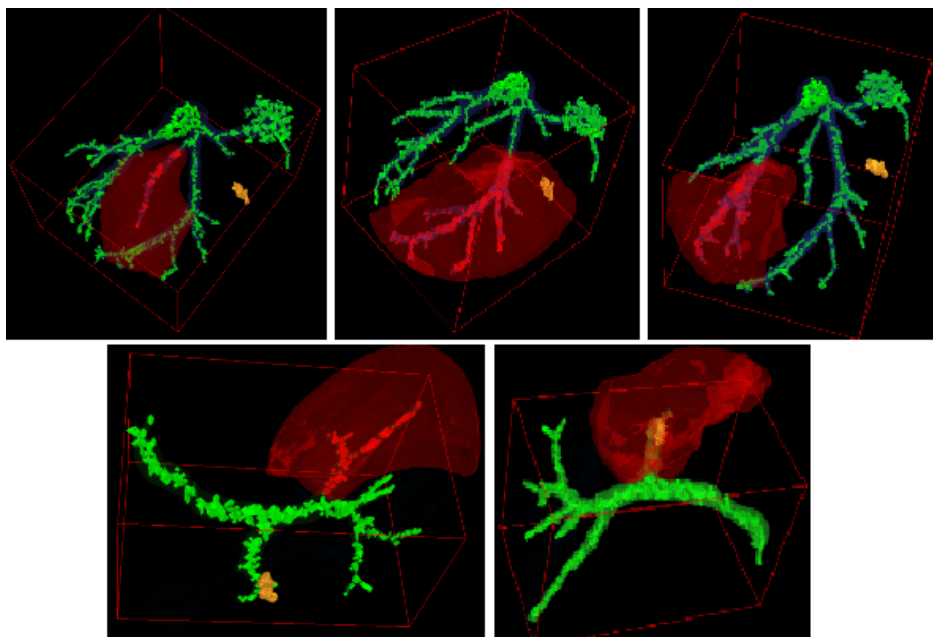
By Lemma 2, the computation in Line 22 takes  $\mathcal{O}(c_i^2 \times (n \times |N| + n \times n_{max}))$  operations, where  $c_i^2$  is the maximum number of cells in a supercell,  $i = 2$  for the 2D case, and  $i = 3$  for the 3D case. By the definition of supercells, we have that  $c_i^2 = m^i$ , for  $i = 2, 3$ . Thus, the computation in Line 22 takes  $\mathcal{O}(m^i \times n \times (|N| + n_{max}))$  operations, where  $i$  is 2 or 3 for the 2D or 3D case, respectively. Recall that  $|N| = 20$  in the 2D case and  $|N| = 80$  in the 3D case. Thus, we have the following

**Theorem 1.** *The runtime of Algorithm 3 is in  $\mathcal{O}(m^i \times n \times n_{max})$ , where  $n_{max}$  is the maximum value of all  $|N(v, 3, M_{R_v})|$ 's,  $R_v$  is defined in Line 6 in Procedure 2, and  $i = 2$  for the 2D case, and  $i = 3$  for the 3D case.*

Exact Euclidean Distance Transform takes  $\mathcal{O}(m^i \times n^i)$  operations [2,6]. Thus, the main algorithm (Algorithm 3) may be faster or slower than the exact Euclidean Distance Transform depending on the value of  $n_{max}$  which depends on the distribution of input type- $i$  cells, for  $i = d, h, l$ .

## 5 Experimental Results

Our experiments used a liver model of  $10^7$  cells within a cuboid which is 324 pixels long, 243 pixel wide, and 129 pixels high. This kind of constant is typical



**Fig. 11.** The diseased liver volumes (i.e., sets  $V$  in Table 2) in the  $i$ -th experiment, from left to right and top to bottom. Note that the tumour was not included in set  $V$  in Experiments 1, 3 and 4. This is because the diseased vein  $S_2$  in Table 2 is only simulated in the experiments.

**Table 2.** By  $i$  we denote the index of an experiment.  $V$  is the volume of the diseased liver,  $S_2$  are the cells inside the volume of the diseased vein, and  $S_1$  are the cells inside of the volume of the healthy vein.

$i$	$V$	$S_2$	$S_1$	$i$	$V_1$	$S_2$	$S_1$
1	87.491	105	2120	4	251.6717	182	1487
2	350.4517	525	1700	5	232.631	153	1516
3	242.179	331	1894				

for a current CT scan of a liver. Each voxel is not perfectly cubic, having side length 0.683 in two directions and 1.0 in the third. We used a PC with 2.50 GHz CPU and 3.0 Gb RAM.

Table 1 shows the relationship between parameter  $m$  as applied in Line 1 of Algorithm 3 and the running time. Times for  $m = 0$  are the running times of the improved brute-force algorithm (Algorithm 2). The experiments indicate that Algorithm 3 is better than Algorithm 2 for  $m = 10$  and  $= 20$ . The algorithm appears to be inefficient if  $m$  is either too small or too large.

Table 2 shows diseased volumes in five experiments. See Fig. 11 for sets  $V$  as mentioned in the table.

## 6 Conclusions

We presented a simple and time-efficient algorithm for separating liver cells using basic ideas of digital geometry. In contrast to existing liver-surgery planning algorithms, our algorithm is not only independent of a surgeons' personal interactive manipulations, but also outputs the exact solution. The paper introduced an important existing problem to the digital geometry community.

**Acknowledgements.** The authors thank all three anonymous reviewers for very valuable comments which have been taken into account for the final paper.

## References

1. Bosch, F.X., Ribes, J., Diaz, M., Cléries, R.: Primary liver cancer: worldwide incidence and trends. *Gastroenterology* 127, S5–S16 (2004)
2. Cao, T.-T., Tang, K., Mohamed, A., Tan, T.-S.: Parallel banding algorithm to compute exact distance transform with the GPU. In: *Symp. Interactive 3D Graphics*, pp. 83–90 (2010)
3. Konrad-Verse, O., Preim, B., Littmann, A.: Virtual resection with a deformable cutting plane. In: *Simulation und Visualisierung*, pp. 203–214 (2004)
4. Klette, R., Rosenfeld, A.: *Digital Geometry*. Morgan Kaufmann, San Francisco (2004)
5. Lamata, P., Lamata, F., Sojar, V., Makowski, P., Massoptier, L., Casciaro, S., Ali, W., Stüdeli, T., Declerck, J., Jackov Elle, O., Edwin, B.: Use of the resection map system as guidance during hepatectomy. *Surg. Endosc.* 24, 2327–2337 (2010)
6. Maurer, C.R., Qi, R., Raghavan, V.: A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions. *IEEE Trans. Pattern Analysis Machine Intelligence* 25, 265–270 (2003)
7. Meinzer, H.-P., Thorn, M., Cordenas, C.E.: Computerized planning of liver surgery – an overview. *Computers & Graphics* 26, 569–576 (2002)
8. Park, H., Bland, P.H., Meyer, C.R.: Construction of an abdominal probabilistic atlas and its application in segmentation. *IEEE Trans. Medical Imaging* 22, 483–492 (2003)
9. Pianka, F., Baumhauer, M., Stein, D., Radeleff, B., Schmied, B.M., Meinzer, H.-P., Müller, S.A.: Liver tissue sparing resection using a novel planning tool. *Langenbecks Arch. Surg.* 396, 201–208 (2011)
10. Reitinger, B., Bornik, A., Beichel, R., Schmalstieg, D.: Liver surgery planning using virtual reality. *IEEE Computer Graphics Applications* 26, 36–47 (2006)
11. Rosenfeld, A., Pfaltz, J.L.: Distance functions on digital pictures. *Pattern Recognition* 1, 33–61 (1968)
12. Shevchenko, N., Seidl, B., Schwaiger, J., Markert, M., Lueth, T.C.: MiMed Liver: A planning system for liver surgery. In: *Int. Conf. IEEE EMBS*, pp. 1882–1885 (2010)
13. Yamanaka, J., Saito, S., Fujimoto, J.: Impact of preoperative planning using virtual segmental volumetry on liver resection for hepatocellular carcinoma. *World J. Surg.* 31, 1249–1255 (2007)



# $O(n^3 \log n)$ Time Complexity for the Optimal Consensus Set Computation for 4-Connected Digital Circles

Gaelle Largeteau-Skapin, Rita Zrour, and Eric Andres

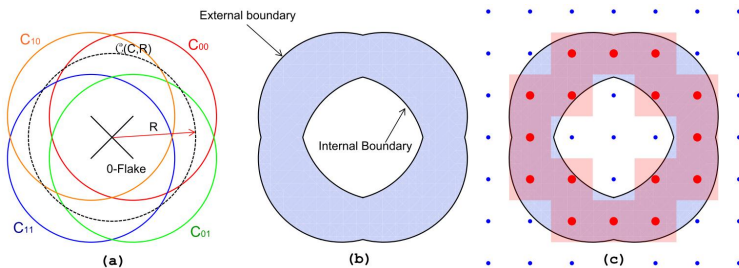
Laboratoire XLIM-SIC UMR CNRS 7252, Université de Poitiers,  
Bld Marie et Pierre Curie, BP 30179, 86962, Futuroscope Chasseneuil Cedex, France

**Abstract.** This paper presents a method for fitting 4-connected digital circles to a given set of points in 2D images in the presence of noise by maximizing the number of inliers, namely the optimal consensus set, while fixing the thickness. Our approach has a  $O(n^3 \log n)$  time complexity and  $O(n)$  space complexity,  $n$  being the number of points, which is lower than previous known methods while still guaranteeing optimal solution(s).

**Keywords:** Shape fitting, consensus set, inliers, outliers, digital circle, 4-connected digital circle, 0-Flake digital circle.

## 1 Introduction

In the present paper, we are considering the fitting problem of a set of points in a noisy 2D image by a 4-connected digital circle. Such a 4-connected digital circle (see Fig. 1(c)) can be obtained by a morphological based digitization scheme. The 0-Flake in Fig. 1(a) is the structuring element. Such circles can be characterized analytically [5,9]. The 0-Flake digital circle is defined as all the digital points (see Fig. 1(c)) inside a sort of Annuli (see Fig. 1(b)), called 0-Flake Annuli, composed



**Fig. 1.** (a) 0-Flake circle and Boundary circles, (b) 0-Flake annulus and (c) corresponding 4-connected digital circle

of four circles (see Fig. 1(a)). These circles are called boundary circles. It is important to note that the thickness of the digital 0-Flake circles is fixed. Most annuli fitting methods consider only classical annuli defined by two concentric circles and try to find annuli with minimal or maximal thicknesses given some other parameters. This is not adequate for digital circle fitting.

The set of points (inliers) which fits a model is called a consensus set. The idea of using such consensus sets was proposed for the RANdom Sample Consensus (RANSAC) method [6], which is widely used in the field of computer vision. However RANSAC is inherently probabilistic in its approach and does not guarantee optimality. This paper aims at proposing a new lower time complexity for the computation of the optimal consensus set. This means that our goal is to maximize the number of inliers. In our case, an inlier is simply defined as a point inside the 0-Flake annulus. Non Probabilistic methods that detect annuli have been proposed (for example [11]). Most of these algorithms minimize or maximize the thickness of the annuli [8] which is not adequate when considering digital circles where the thickness is fixed. Only few algorithms deal with outliers [12,7,13] but the number of outliers is usually predefined [7,13] and the problem consists again in minimizing the thickness. The method proposed by O'Rourke et al. [14,15] that transforms a circle separation problem into a plane separability problem, is also not well suited because the fixed thickness of the digital circles translates into non fixed vertical thicknesses for the planes. In this case, the problem is complicated (See [4] for some solutions on how to handle this difficulty).

So our problem is finding the optimal consensus set (maximal number of inliers) of digital points inside a 0-Flake annulus which has a fixed thickness where the center and the radius are unknowns. In [1] and [2], brute force algorithms were proposed to compute the optimal consensus set respectively for Andres digital circle (defined as digital points inside a classical annulus of fixed thickness) and 0-Flake digital circles. It was shown that if an optimal solution exists then there exists a finite number of equivalent optimal solutions (with the same set of inliers) with three points on the boundary (internal and/or external) of the annulus. Testing all the configurations of three points and counting the inliers leads therefore to all the possible optimal solution sets with a time complexity of  $O(n^4)$  where  $n$  is the number of points.

A new method is proposed in this paper for fitting 0-Flake digital circles. This method requires just two points to be located on the boundary circles. This method is inspired by the dual space proposed by [11]: the centers of all the circles with two specific points on the boundary correspond to a straight line. A dual space where the  $x$  axis represents the center locations and the  $y$  axis represents the distance to this center allows to find the largest empty annulus using an interval sorting.

We adapt this idea to our problem but there are several major differences: in our case, the thickness is fixed and we look for a maximal number of inliers. Moreover, since we deal with Flake annuli, there are more than only one straight line for the center locations (see section 2.2). The idea is the following: given a set

$S$  and given two specific points on the boundary of a Flake annuli, we consider the center locations straight lines as parametric axis. We then determine when a point enters and leaves the flake annuli while the center moves along the axis. This allows us to compute the intervals where the number of inliers is maximized (Section 3). By considering all the combinations of two points, we are able to compute the exhaustive set of all optimal consensus sets in  $O(n^3 \log n)$ .

The paper is organized as follows: in Section 2 we expose some properties and characterizations of the 0-Flake digital circles and its analytical annulus definition. Section 3 provides the general idea and the detailed algorithm for finding the optimal consensus sets. Section 4 presents some results. Finally Section 5 proposes a conclusion and some perspectives.

## 2 The 0-Flake Annulus : Definitions and Properties

In [1] and [2], we proposed a brute force algorithm with  $O(n^4)$  time complexity for fitting Andres circle and 0-Flake circle of fixed thickness, with  $n$  the number of points to fit. We have shown that if an optimal solution (set of inliers) exists then there exists an equivalent optimal solution (with the same set of inliers) with three points on the boundary (internal and/or external). In this section we are considering the problem of characterizing the 0-Flake annulus that are equivalent (same inliers, same thickness) to some optimal solution with only two points on the boundary circles of the Flake annuli. Let us first introduce some basic notations as well as the analytical definition of the 0-Flake digital circles. In a second part of this section, we will look at the annulus characterization for 0-Flake circles with thickness 1.

### 2.1 Notations and Basic Definitions

In this section, we present 0-Flake digital circles with the associated notations and definitions. See [10,5,3,9] for more details on the digitization models and properties of the different types of digital circles. The digitization scheme we are considering is an *Adjacency Flake Digitization* [5,3,9]. It is based on a morphological digitization scheme with a structuring element called an *Adjacency Flake*. In this paper we are limiting our self to 0-adjacency Flake (or simply 0-Flake) circles for the sake of simplicity. The 2D 0-Flake corresponds to the diagonals of a unit cube. The figure 1.a shows the 0-Flake and a corresponding Flake annulus. This corresponds to 4-connected digital circles when the size of the Flake and thus the thickness of the Flake annulus is equal to one. However, the proposed fitting method works as well for 2D 1-adjacency Flake circles (8-connected circles) and for other thicknesses [3,9]. The 0-Flake digitization  $D_{F_0}$  of the Euclidean circle  $\mathcal{C}(C, R)$  of center  $C$  and radius  $R$  is defined as follows:

$$D_{F_0}(\mathcal{C}(C, R)) = (X \otimes \mathcal{C}(C, R)) \cap \mathbb{Z}^2$$

Where  $X$  is the 0-Flake corresponding to the diagonals of a unit square. Proof that such a digital circle is 4-connected can be found in [5,9].

The 0-Flake annulus  $A_{F_0}$  of the Euclidean circle  $\mathcal{C}(C, R)$  of center  $C$  and radius  $R$  is analytically defined as follows [9]:

$$A_{F_0}(\mathcal{C}(C, R)) = (X \otimes \mathcal{C}(C, R)) = \left\{ x \in \mathbb{R}^2 : -|x - C_x| - |y - C_y| - \frac{1}{2} \leq (x - C_x)^2 + (y - C_y)^2 - R^2 \leq |x - C_x| + |y - C_y| + \frac{1}{2} \right\}$$

The smallest possible 0-Flake circle is of radius  $\sqrt{2}/2$ . With a Flake structuring element, the analytical characterization of circles of smaller radii are not correct. This is one of the limitations of the Flake model. It is however not a big constraint as it corresponds to a circle that spans only a couple of pixels [9].

We call **boundary circles** the 4 circles that form the boundary of the 0-Flake annulus, i.e. the circles centered on  $(C_x \pm \frac{1}{2}, C_y \pm \frac{1}{2})$ . On figure 1.a, we can see the four boundary circles  $C_{00}$ ,  $C_{01}$ ,  $C_{10}$  and  $C_{11}$ :

**Definition 1.** Let  $\mathcal{C}_{ij}$  be a boundary circle of the 0-Flake circle  $\mathcal{C}(C_x, C_y)$  of radius  $R$ .  $\mathcal{C}_{ij}$  is defined as the circle of center  $(C_x, C_y) + (1/2, 1/2) - (i, j)$  and radius  $R$ .

The actual boundaries of the 0-Flake annulus are only parts of those boundary circles (see fig. 1). We call internal (resp. external) boundary of the 0-Flake annulus, the parts of the boundary circles that are closest (resp. farthest) to the center of the 0-Flake annulus. We define the 0-Flake digital circle as the set of digital points in the 0-Flake annulus.

**Definition 2.** Let us consider a set of points  $S$ . Two Flake annuli are said to be equivalent with regard to  $S$  if the points of  $S$  belong to both annuli.

## 2.2 0-Flake Annulus Characterization

In [2], it has been proven that given a 0-Flake annulus covering a set of points there exists an equivalent 0-Flake annulus (same inliers, same thickness) which has at least three points of the set on its boundary circles, not necessarily on the actual internal or external boundary of the annulus. For the fitting method we are going to present here, we need to show that if a Flake annulus covers a set of points then there exists an equivalent Flake annulus with two points on the boundary circles. Again, we do not require the points to be on the actual internal or external boundary as simply being on the boundary circles is sufficient to provide a straight line of possible center locations. The proof in [2] is obviously sufficient for our purpose.

Now we have two points of the set on the boundary circles, we are going to check all the possible 0-Flake annuli that have those two points on their boundary circles. The following proposition provides the characterization of the center locations for those annuli.

**Proposition 1.** Let us suppose that we have a set of points  $S$  and two points  $p$  and  $q \in S$ .

- The centers of all the Flake annuli having  $p$  and  $q$  on their boundary circles belong to a maximum of 16 straight lines. We call these straight lines, center axes.
- The set of all the centers of the annuli covering the consensus set  $S$  with  $p$  and  $q$  on the boundary circles is a set of straight line segments, half lines or straight lines belonging to the **center axes**.

*Proof.* Let us suppose that we have a consensus set  $S$  in a 0-Flake annulus  $F$  of center  $C(C_x, C_y, R,)$  with two points  $p$  and  $q$  of  $S$  on its boundary circles. First, let us note that if we consider Flake annuli with two points  $p$  and  $q$  on some of its boundary circles, we have several possibilities since  $p$  and  $q$  may belong to the boundary circles  $C_{00}, C_{01}, C_{10}$  or  $C_{11}$ . There are 16 possible different configurations.

- If  $p$  and  $q$  belong to the same boundary circle  $C_{ij}$  of center  $(C_{ijx}, C_{ijy}) = (C_x, C_y) + (i, j) - (1/2, 1/2)$  then the center of  $C_{ij}$  has to belong to the perpendicular bisector of  $p$  and  $q$ . Therefore the 0-Flake annulus  $F$  centers belong to its parallel passing through the points  $(C_{ijx}, C_{ijy}) + (1/2, 1/2) - (i, j)$ .
- If  $p$  belongs to the boundary circle  $C_{ij}$  and  $q$  to the boundary circle  $C_{kl}$  then obviously the point  $q' = q + (i - k, j - l)$  belongs to  $C_{ij}$  and the previous reasoning works with  $p$  and  $q'$  at the condition that  $p$  is different from  $q'$ . In this case there is no center axis but all the points in space can be centers. We can actually discard such configurations because in such a case it is easy to see that there exist an equivalent configuration with two points of  $S$  on the boundary circles that do not have this problem. One has simply to discard one of the two points, for instance by keeping  $p$ , and use the principle of [2] to find another point on a boundary circle. Since only four points around  $p$  may cause such a problem, we have either other points or a set of four neighboring points that can easily be dealt with otherwise. The corresponding optimal solutions will therefore be treated by some other configurations of points.

This proves that there is a maximum of 16 center axes. The actual center axis equation for  $p \in C_{ij}$  and  $q \in C_{kl}$  is given by:

$$2(p_x - q_x + i - k)C_x + 2(p_y - q_y + j - l)C_y + (p_x - 1/2 + i)^2 + (p_y - 1/2 + j)^2 - (q_x - 1/2 + k)^2 - (q_y - 1/2 + l)^2 = 0$$

Let us now consider a point  $t$  of  $S$  and a center axis defined by  $p$  and  $q$  belonging to  $C_{ij}$  and  $C_{kl}$  respectively, with  $i, j, k, l \in \{0, 1\}$ . The point  $t$  is an inlier if it is inside of at least one of the four boundary circles and not inside all four boundary circles (see Figure 1.a). Let us determine when, with the center of the flake annulus moving along the center axis,  $t$  enters or leaves a boundary circle  $C_{mn}$  and thus when it may be inside 0,1,2,3 or 4 boundary circles. Different cases have to be examined:

- Let us suppose that  $t - (m, n) \neq p - (i, j)$  and  $t - (m, n) \neq q - (k, l)$ . Let us consider the following three points  $p, q' = q + (i - k, j - l)$  and

$t' = t + (i - m, j - n)$ . There is of course only one circumcenter  $c$  for  $p, q'$  and  $t'$ . There is therefore a unique center point  $c' = c + (1/2, 1/2) - (i, j)$  on the center axis such that  $p \in C_{ij}$ ,  $q \in C_{kl}$  and  $t \in C_{mn}$ . On one side of  $c'$  on the center axis,  $t$  will be inside the boundary circle  $C_{mn}$  while for all the centers on the other side of  $c'$  on the center axis,  $t$  will be outside the boundary circle  $C_{mn}$ .

- If  $t - (m, n) = p - (i, j)$  or  $t - (m, n) = q - (k, l)$  then  $t$  belongs to  $C_{mn}$  for all the Flake annuli with center on the center axis.

Now, for each center axis defined by  $p$  and  $q$ , with some parametrization of the center axis, for each point  $t$  we have four intervals of type  $]-\infty, x]$ ,  $[x, +\infty[$  or  $]-\infty, +\infty[$ . The intersection of four such intervals is a straight line segment on the center axis, an half-line on the center axis, the complete center axis or is empty. By considering all the center axes we obtain the result of the proposition.  $\square$

### 3 Fitting Algorithm

Using the above proposed flake annuli characterization, our fitting problem can be described as follows: given a finite set  $S = \{(P_x, P_y) \in \mathbb{Z}^2\}$  of  $n$  points such that  $n \geq 2$ , and given a fixed thickness 1 we would like to find a 0-Flake annulus such that it contains the maximum number of points of  $S$ . Points belonging to the annulus are called inliers; otherwise they are called outliers.

The idea behind our fitting method is inspired by [11] where the authors try to maximize the width of an empty annulus. In [11], given two points  $p$  and  $q$ , they define a dual space where the perpendicular bisector of the two points becomes the abscissa axis. These are all the centers of the circles that have  $p$  and  $q$  on its boundary. For any point  $t$ , the ordinate value is given by its distance to a point of the bisector and thus to the center of a circle that has  $p$  and  $q$  on its boundary. It allows them to determine when a point  $t$  enters a circle centered on the bisector. By sorting these entry points relatively to the abscissa axis, they determine the biggest empty annulus. Since they look for the biggest empty annulus, they do not represent an annulus in their parameter space but only circles. It is the biggest empty interval projected on the abscissa axis that will define the looked for annulus.

Our purpose is quite different but their idea of taking the axes where the possible centers of the annuli are located can be adapted in the following way. One of the main difference with our problem is that we deal with Flake annuli and therefore we have four boundary circles that are not concentric: for two given points  $p$  and  $q$ , the 0-Flake annuli centers may follow 16 different straight lines, called **center axis**. Each of these center axis will be considered separately. We do not consider an actual dual space. There is no ordinate axis since the distance from  $p$  and  $q$  to the center axes and thus to the center of the Flake annuli are not equal. We simply consider a parametrization of each center axis and determine

the parameter  $\lambda_{ij}^t$  for which a given point  $t$  enters the boundary circle  $C_{ij}$  of a flake annulus centered on the considered center axis. This corresponds to a set of a maximum 4 parameter values that are sorted. We complete this list by adding  $-\infty$  and  $+\infty$ . Each parameter interval (between two consecutive values of the list) is tested to check if the point  $t$  is inside or outside the corresponding 0-Flake annulus. One needs only to test the midpoint of each interval and in the case of semi open intervals such as  $]-\infty, \lambda]$  (respectively  $[\lambda, +\infty[$ ), we test a value that is significantly smaller (respectively bigger) than  $\lambda$  in order to avoid numerical problems. This leads to a set of one or two intervals where  $t$  is inside

---

**Algorithm 1.** 0-Flake annulus fitting

---

```

input : A set S of  $n$  grid points
output: A list V of centers and radius values for the best fitted 0-Flake annuli
1 begin
2   initialize  $Max = 0$ ;
3   initialize the list V to the empty list;
4   foreach  $p \in \mathbf{S}$  do
5     foreach  $q \in \mathbf{S}$  do
6       foreach of the 16 different configurations of  $p$  and  $q$  do
7         compute the straight line  $\Delta_{pq}$  where the center are located;
8         initialize the list of parameters  $L_\lambda$ ; foreach  $t \in \mathbf{S}$  do
9           initialize the valide interval to  $]$ ;
10          foreach For each one of the four boundary circle : do
11            compute the parameter  $\lambda$  for which the point  $t$  is ON
12            the boundary circles;
13            test a value in the interval  $]\infty, \lambda]$  to know if the point is
14            inside the boundary circle for this interval;
15            keep the valid interval  $I_t$  where  $t$  is inlier with the
16            following rule: when  $t$  belongs to the zero or four circles
17            it is an outlier;
18          foreach sub-interval  $[min, max]$  in  $I_t$  do
19            Add the couples  $(min, 1)$  and  $(max, -1)$  to the
20            parameter list  $L_\lambda$ ;
21          Sort the pair elements  $(\lambda_k, f_k)$  of  $L_\lambda$  with the values  $\lambda_k$  as keys;
22          Initialize  $F = 0$ ;
23          foreach couple  $(\lambda_k, f_k)$  in  $L_\lambda$  do
24             $F = F + f_k$ ;
25            if  $F > Max$  and  $f_{k+1} = -1$  then
26              Set  $Max = F$ ; Erase V and set it to the interval
27               $[\lambda_k, \lambda_{k+1}]$  ;
28            if  $F = Max$  and  $f_{k+1} = -1$  then
29              Add the interval  $[\lambda_k, \lambda_{k+1}]$  to V
30          return V;

```

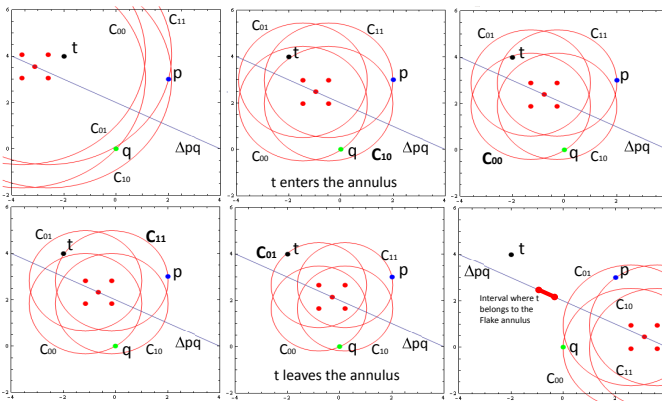
---

a 0-Flake. In order to avoid interval sorting (which might have a  $O(n^2)$  worst case complexity), these intervals are then simply coded as a general parameter list as follows: let us suppose that the point  $t$  belongs to a 0-Flake annulus for the interval  $[a, b]$ , then we add the elements  $(a, +1)$  and  $(b, -1)$  to a *general parameter list*. This codes for the fact that at parameter value  $a$  the number of inliers is increased by 1 and at parameter value  $b$  it is decreased by 1.

This is repeated for each point  $t$  (different from  $p$  and  $q$ ) of the set  $S$ . The general parameter list is then sorted by parameter value and, starting at parameter value  $-\infty$ , the number of inliers are counted by summing up the  $+1$  and  $-1$ . This results in a list of intervals for which a maximum of inliers is obtained. These intervals and their corresponding generator values  $p, q, i, j, k$  and  $l$  are added to the already existing maximal inlier interval list. If the maximum of inliers increases, then the former maximal inlier interval list is wiped and replaced by a new one.

A pair of points defines a maximum of 16 center axes. For each other point, we determine the parameter values on each center axis for which it is inside a flake annulus. There is a maximum of 4 such parameter values per point. All these are sorted for each center axis with a time complexity of therefore  $n \log n$ . scanning each list to determine the interval where we have a maximum consensus set is linear in the number of parameter values and thus in  $n$ . Since this is repeated for every couple of points in the set, the final complexity is  $O(n^3 \log n)$ .

**Example:** Here is an example of values obtained while fitting the points  $(0, 0)$ ,  $(5, 3)$  and  $(2, 1)$ . At some point we have a (already sorted) set of parameter values  $-23.4953, -4.0588, -3.08697, 8.57493$  for a center axis corresponding to  $(0, 0)$  and  $(5, 3)$ . The corresponding general parameter list looks like  $((-\infty, 1), (-3.08697, -1), (8.57493, 1), (+\infty, -1))$ . This means that the point  $t$  (in this case  $(2, 1)$ ) belongs to a 0-Flake annulus for the parameter intervals  $]-\infty, -3.08697]$  and  $[8.57493, +\infty[$ . The parameter values  $-23.4953$  and  $-4.0588$



**Fig. 2.** The 0-Flake annulus with  $p \in C_{11}$  and  $q \in C_{00}$  and an interval where the point  $t$  belongs to the Flake annulus

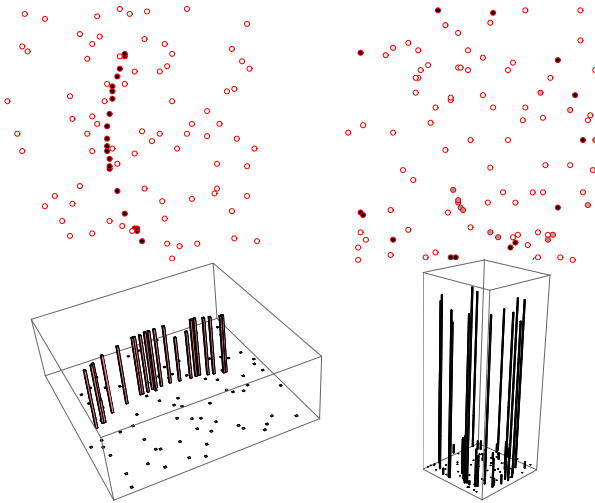


disappear as they correspond to  $t$  leaving or entering a boundary circle inside the annulus. Note that  $(+\infty, -1)$  is not really needed for the inlier computation but it is useful for expressing the intervals.

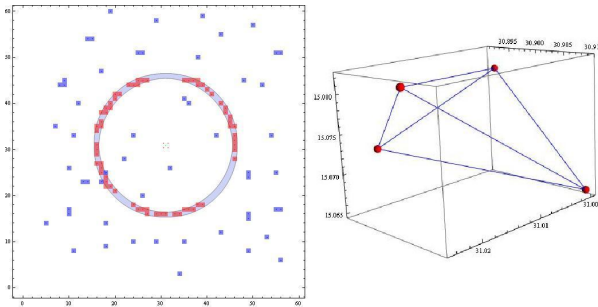
Fig. 2 gives an example of a 0-Flake annulus with  $p \in C_{11}$  and  $q \in C_{00}$ . Doing this for all the couple of points among the set of points to fit yields the optimal 0-Flake annulus in terms of number of inliers.

### 4 Experiments

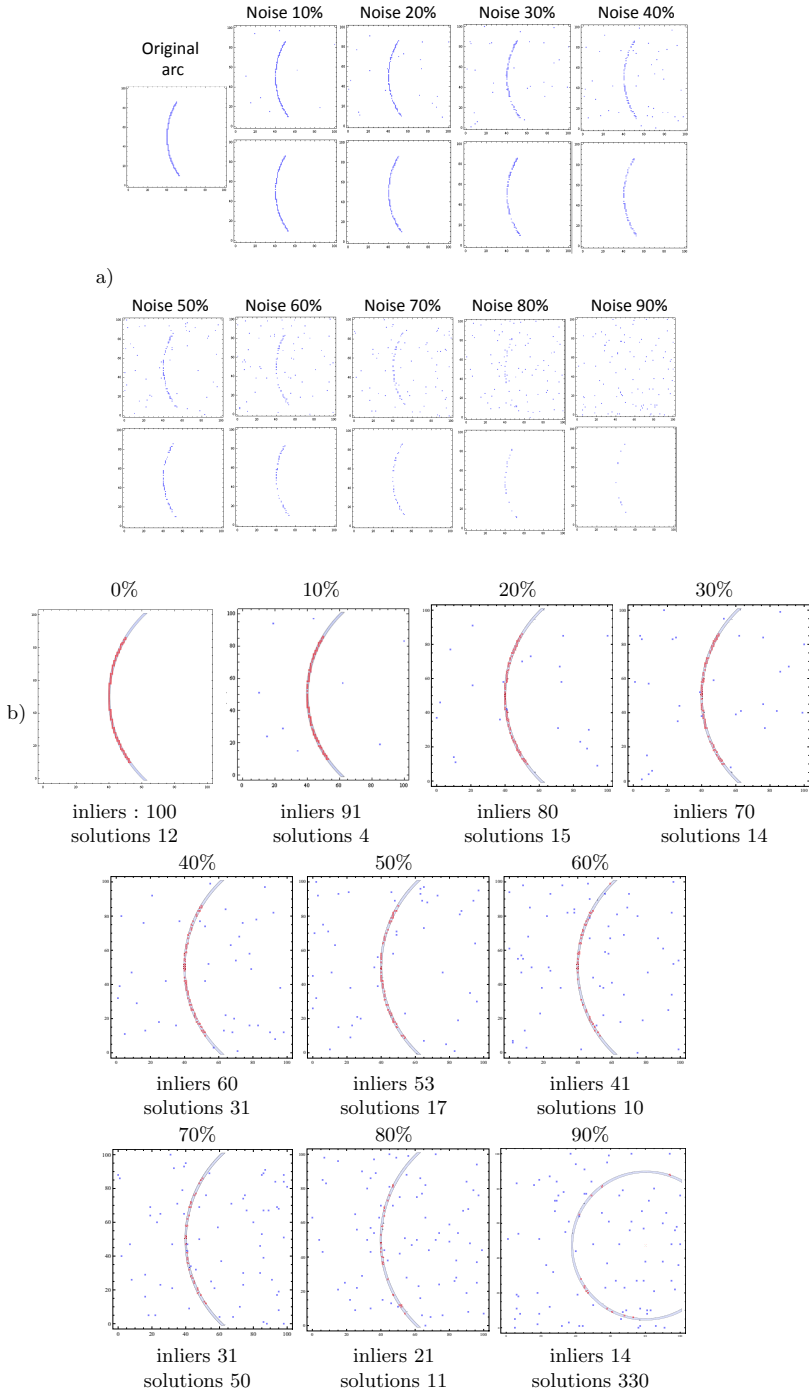
We used Mathematica for implementing our method. We applied our method for 2D noisy 0-Flake annuli as shown in fig. 4(a). A bounding region (center,radius)



**Fig. 3.** A representation of the number of times a point belongs to the optimal consensus sets found for, on the left side an example with 85% noise and on the right side an example with 90% noise



**Fig. 4.** a) Fitting of 2D noisy 0-Flake circles. b) bounding region of all the optimal centers.



**Fig. 5.** Tests with different levels of noise on a digital 0-Flake circular arc

of all the possible solutions corresponding to optimal consensus sets for this image are shown in fig. 4(b).

The figure 5 presents a 0-Flake circular arc of 100 points and some degraded versions of it: In each image, we keep 100 points but for the different degraded versions of  $x\%$  noise, we kept  $100 - x$  random points of the original arc and added  $x$  randomly located points as noise in the background. Part a) presents the tested data : the noisy arc with  $x$  points (bottom) and this arc and its noisy background (top). Part b) shows one solution for each case, the optimal number of inliers found and the number of distinct limit solutions (corresponding to an center interval end point).

## 5 Conclusion and Perspectives

In this paper we have presented a new method for fitting 0-Flake digital circles to a set of points while fixing the thickness. Various papers have been written on fitting circles or annuli but usually they have not dealt with fixed thicknesses which is a fundamental property of digital circles. Our approach guarantees optimal results from the point of view of maximal consensus sets: we are guaranteed to fit a digital circle with the least amount of outliers. In terms of computation time, this approach has a lower time complexity than the one presented in [2]. The method is general enough that it can be extended to 1-Flake circles, Andres circles and probably most other types of digital circles [3,9] with thicknesses not limited to 1. This work opens many interesting perspectives for the future. One obvious question that remains open is the question of the optimal time complexity we can expect for such a problem. We have reasons to believe that we can not beat a  $O(n^3)$  time complexity simply because this is the optimal time complexity for a similar problem of 3D plane fitting [16]. Now, the reason why we suspect that the optimal time complexity might be the same is simply because of some arguments coming from conformal space representations. This needs however to be proved and an according method would need to be found. One of the interesting aspects that has not been yet fully explored is that computing optimal consensus sets or near optimal consensus sets allows us to classify points and introduce notions of **strong** or **weak** inliers . We can for instance, differentiate inliers that belong to many optimal or near optimal consensus sets from points that only belong to some of those solutions (See Figure 6. for an example). The method we proposed seems to extend pretty well to higher dimension but we need a formal proof of the Flake annulus characterization in higher dimensions. A last perspective is of course fitting of other types of digital curves such as digital conics for instance.

**Acknowledgments.** The work for this paper was partly financed by Egide, franco-Japanese PHC Sakura project n 27608XJ and by the Poitou Charentes region project n 11/RPC-R-051.

## References

1. Zrou, R., Largeteau-Skapin, G., Andres, E.: Optimal Consensus Set for Annulus Fitting. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) DGCI 2011. LNCS, vol. 6607, pp. 358–368. Springer, Heidelberg (2011)
2. Andres, E., Largeteau-Skapin, G., Zrou, R., Sugimoto, A., Kenmochi, Y.: Optimal Consensus Set and Preimage of 4-connected circles in a noisy environment. In: 21st International Conference on Pattern Recognition. IEEE Xplore, Tsukuba Science City (2012)
3. Andres, E., Roussillon, T.: Analytical Description of Digital Circles. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) DGCI 2011. LNCS, vol. 6607, pp. 235–246. Springer, Heidelberg (2011)
4. Phan, M.S., Kenmochi, Y., Sugimoto, A., Talbot, H., Andres, E., Zrou, R.: Efficient Robust Digital Annulus Fitting with Bounded Error. In: Gonzalez-Diaz, R., Jimenez, M.-J., Medrano, B. (eds.) DGCI 2013. LNCS, vol. 7749, pp. 253–264. Springer, Heidelberg (2013)
5. Fiorio, C., Toutant, J.-L.: Arithmetic Discrete Hyperspheres and Separatingness. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 425–436. Springer, Heidelberg (2006)
6. 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)
7. Har-Peled, S., Wang, Y.: Shape Fitting with Outliers. *SIAM Journal on Computing* 33(2), 269–285 (2004)
8. Rivlin, T.J.: Approximation by circles. *Computing* 21(2), 93–104 (1979)
9. Toutant, J.-L., Andres, E., Roussillon, T.: Digital Circles, Spheres and Hyperspheres: From Morphological Models to Analytical Characterizations and Topological Properties. Submitted to *Discrete Applied Mathematics Journal* (2012)
10. Andres, E., Jacob, M.-A.: The Discrete analytical hyperspheres. *IEEE Transactions on Visualization and Computer Graphics* 3, 75–86 (1997)
11. Diaz-Banez, J.M., Hurtado, F., Meijer, H., Rappaport, D., Sellarès, T.: The largest empty annulus problem. *International Journal of Computational Geometry and Applications* 13(4), 317–325 (2003)
12. Dunagan, J., Vempala, S.: Optimal outlier removal in high-dimensional spaces. *Journal of Computer and System Sciences* 68(2), 335–373 (2004)
13. Matousek, J.: On enclosing  $k$  points by a circle. *Information Processing Letters* 53(4), 217–221 (1995)
14. O’Rourke, J., Kosaraju, S.R., Megiddo, N.: Computing circular separability. *Discrete and Computational Geometry* 1, 105–113 (1986)
15. 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)
16. Aiger, D., Kenmochi, Y., Talbot, H., Buzer, L.: Efficient Robust Digital Hyperplane Fitting with Bounded Error. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) DGCI 2011. LNCS, vol. 6607, pp. 223–234. Springer, Heidelberg (2011)

# Efficient Robust Digital Annulus Fitting with Bounded Error<sup>\*</sup>

Minh Son Phan<sup>1,2</sup>, Yukiko Kenmochi<sup>1</sup>, Akihiro Sugimoto<sup>3</sup>, Hugues Talbot<sup>1</sup>,  
Eric Andres<sup>4</sup>, and Rita Zrouf<sup>4</sup>

<sup>1</sup> Université Paris-Est, LIGM, UPEMLV-ESIEE-CNRS, France

<sup>2</sup> Université de Strasbourg, LSIIT, France

<sup>3</sup> National Institute of Informatics, Japan

<sup>4</sup> Laboratory XLIM, SIC Department, University of Poitiers, France

**Abstract.** A digital annulus is defined as a set of grid points lying between two circles sharing an identical center and separated by a given width. This paper deals with the problem of fitting a digital annulus to a given set of points in a 2D bounded grid. More precisely, we tackle the problem of finding a digital annulus that contains the largest number of inliers. As the current best algorithm for exact optimal fitting has a computational complexity in  $O(N^3 \log N)$  where  $N$  is the number of grid points, we present an approximation method featuring linear time complexity and bounded error in annulus width, by extending the approximation method previously proposed for digital hyperplane fitting. Experiments show some results and runtime in practice.

**Keywords:** fitting, annulus, approximation, halfspace range searching.

## 1 Introduction

Shape fitting is widely used in computer vision, computational geometry, image analysis and many other areas. Given a set of points  $S$ , a shape model  $M$  (for example, lines, curves, planes, or circles), and some criterion  $F$ , the problem is to fit  $M$  to  $S$  under  $F$ . Depending on the criteria employed, we have several well-known methods dealing with this problem. The method of least-squares [1] minimizes the sum of squared residuals from all given points. This method is popular, but it is not efficient in the presence of outliers. In order to enhance the robustness, many alternative methods have also been proposed, such as the method of least-absolute-values [2], the method of least-median-of-squares [3]. However, these methods are still not robust in the presence of many outliers. One of the commonly used methods for robust shape fitting is RANdom SAMple Consensus (RANSAC) [4]. It can provide robust results even if there is a significant number of outliers, with reasonable runtime in practice, while it provides neither guarantee of optimal solution nor bound of approximation error.

---

<sup>\*</sup> This work has been partly supported by the French *Agence Nationale de la Recherche* (ANR-2010-BLAN-0205 03) and a French-Japanese joint research project called the Sakura program (No. 27608XJ).

In this paper, we use a criterion similar to that of RANSAC. As we assume that our input data is discrete, such as a digital image, we use a digital shape model, instead of a continuous one as generally used for the above mentioned conventional methods. Fitting of digital lines [5] or digital planes [6] has been already studied. This paper aims at exploring the problem of fitting a digital circle. We consider a digital circle as a digital annulus [7], defined as a set of grid points lying between two circles sharing a center but with different radii. This paper treats the problem of fitting a digital annulus to a given set of points in a 2D bounded grid. More precisely, we tackle the problem of finding a digital annulus that contains the largest number of inliers. The current best algorithm for exact optimal fitting has a computational complexity in  $O(N^3 \log N)$  [8]. We present an approximation method featuring linear time complexity and bounded error in annulus width by extending an approximation method proposed for digital hyperplane fitting [6].

We first formulate our digital annulus fitting as the digital plane fitting problem with some particular setting. For our approximation method, we then employ the approximate halfspace range searching technique [9], which consists of two main steps: generating an approximate range space, and providing a data structure to solve approximate halfspace range searching with a constant query time. We follow these two steps, in the context of solving our converted digital plane fitting, by combining two problems of approximate halfspace range searching. We also explain how to interpret results of digital plane fitting in the context of digital annulus fitting, with an interpretation of bounded error. Finally, we provide experimental results and measured runtimes.

## 2 Problem of Digital Annulus Fitting

### 2.1 Digital Annulus and Its Fitting Problem

In this paper, we follow the definition of a digital annulus as introduced by Andres *et al.* [7] and used by Zrour *et al.* for fitting [8]. We first describe the continuous model of circles. A continuous circle  $\mathbf{C}$  with center  $(a, b)$  and radius  $r$  in the Euclidean space  $\mathbb{R}^2$  is defined by

$$\mathbf{C} = \{(x, y) \in \mathbb{R}^2 : (x - a)^2 + (y - b)^2 = r^2\} \quad (1)$$

where  $a, b, r \in \mathbb{R}$ . In the discrete space  $\mathbb{Z}^2$ , the digitization of (1) is defined by a set of grid points lying between two circles that share the same center  $(a, b)$  and two different radii  $r - \frac{w}{2}$  and  $r + \frac{w}{2}$ , where  $w \in \mathbb{R}$  is the width between the two circles. Namely,

$$\mathbf{A} = \left\{ (x, y) \in \mathbb{Z}^2 : \left( r - \frac{w}{2} \right)^2 \leq (x - a)^2 + (y - b)^2 \leq \left( r + \frac{w}{2} \right)^2 \right\}. \quad (2)$$

This is called a *digital annulus*, or Andres circle [7]. Given a set  $S$  of discrete points coming from a  $[0, \delta]^2$  grid, where  $\delta \in \mathbb{N}$ , the problem is to fit this digital

annulus model  $\mathbf{A}$  to  $S$  such that the fitted digital annulus contains as many points of  $S$  as possible, called inliers. This problem is described below. Points that are not contained in the model are called outliers.

**Problem 1 (Digital annulus fitting)**

*Input* : A set  $S$  of discrete points bounded in a  $[0, \delta]^2$  grid, and a fixed width  $w$ .

*Output* : A digital annulus  $\mathbf{A}$  containing the maximum number of inliers in  $S$ .

This digital annulus fitting problem is solved by an exact method, introduced in [8], in time  $O(|S|^3 \log |S|)$ . This complexity of the exact solution is not feasible in practice when  $|S|$  is large. In some cases it may be sufficient to compute an approximate solution if the solution is obtained with inexpensive computational complexity. We thus propose here an approximate method with linear runtime and bounded error, instead of searching for the exact solution. For this purpose, we first convert our digital annulus fitting problem into a more simple problem in the following.

**2.2 Digital Plane Fitting Induced by Digital Annulus Fitting**

Since the inequalities (2) are non-linear with respect to the unknown parameters  $a, b, r$ , we convert those to linear ones along the known conversion [10]. Then, the digital annulus formula of (2) can be converted into the digital plane formula [11], defined by two inequalities of the form  $0 \leq Ax + By + z + C \leq W$  where  $W$  is the width.

The inequalities (2) are written as:

$$\left(r - \frac{w}{2}\right)^2 \leq -2ax - 2by + x^2 + y^2 + a^2 + b^2 \leq \left(r + \frac{w}{2}\right)^2. \tag{3}$$

Letting

$$\begin{cases} z = x^2 + y^2, \\ A = -2a, \\ B = -2b, \\ C = a^2 + b^2 - \left(r - \frac{w}{2}\right)^2, \\ C' = a^2 + b^2 - \left(r + \frac{w}{2}\right)^2, \end{cases} \tag{4}$$

we have that

$$0 \leq Ax + By + z + C \leq C - C'.$$

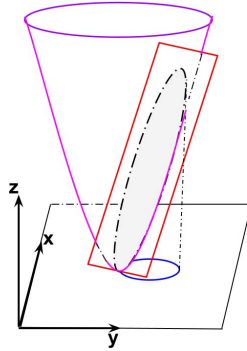
We can thus define the converted digital plane model as follows:

$$\mathbf{P} = \{(x, y, z) \in \mathbb{Z}^3 : 0 \leq Ax + By + z + C \leq C - C'\}. \tag{5}$$

Obviously this has the form of a digital plane [11], which is defined as a set of grid points lying between two parallel planes, whose width is set to be  $W = C - C'$ . Therefore, (5) leads to the following observation.

**Observation 1.** *The problem of digital annulus fitting is equivalent to the problem of digital plane fitting for the model defined by (5).*

This converted problem is described as follows.



**Fig. 1.** Illustration of converting a circle (colored in blue) in 2D into a plane (colored in red) in 3D. The converted 3D points are on the parabola  $z = x^2 + y^2$  (colored in violet).

**Problem 2 (Converted digital plane fitting)**

*Input :* A set  $S$  of discrete points bounded in a  $[0, \delta]^2$  grid, and width  $w$ .

*Output :* A digital plane  $\mathbf{P}$  containing the maximum number of inliers when the input set is  $S' = \{(x, y, x^2 + y^2) : (x, y) \in S\}$ .

Figure 1 illustrates the relationship between  $\mathbf{A}$  and  $\mathbf{P}$ , so that the 2D coordinates  $(x, y)$  of  $\mathbf{A}$  are converted into 3D coordinates  $(x, y, x^2 + y^2)$  of  $\mathbf{P}$ . In fact, the setting of new parameters in (4) is not efficient since  $\forall (x, y) \in [0, \delta]^2 \Rightarrow z = x^2 + y^2 \in [0, 2\delta^2]$ , and consequently the 2D grid of size  $[0, \delta]^2$  for digital annulus fitting is changed to the 3D grid of size  $[0, \delta]^2 \times [0, 2\delta^2]$  for the converted digital plane fitting; the grid size for  $z$  is too large in practice. We therefore define, instead of (4),

$$\begin{cases} z = \frac{1}{2\delta}(x^2 + y^2), \\ A = -\frac{a}{\delta}, \\ B = -\frac{b}{\delta}, \\ C = \frac{1}{2\delta} \left( a^2 + b^2 - \left( r - \frac{w}{2} \right)^2 \right), \\ C' = \frac{1}{2\delta} \left( a^2 + b^2 - \left( r + \frac{w}{2} \right)^2 \right), \end{cases} \tag{6}$$

by dividing all the terms in (3) by  $2\delta$ . With this setting, we see that  $x, y \in [0, \delta] \Rightarrow z = \frac{1}{2\delta}(x^2 + y^2) \in [0, \delta]$ , so that we have a 3D grid  $[0, \delta]^3$  for digital plane fitting.

Concerning the ranges of the new parameters  $A, B, C$  and  $C'$ , we first obtain  $a, b \in [0, \delta] \Rightarrow A = -\frac{a}{\delta}, B = -\frac{b}{\delta} \in [-1, 0]$ . Accordingly, the search space for the converted digital plane is reduced, compared with the standard digital plane fitting. To enforce  $\left( r - \frac{w}{2} \right)^2$  and  $\left( r + \frac{w}{2} \right)^2$  to be positive,  $C$  and  $C'$  in (6) must satisfy

$$C, C' \leq \frac{a^2 + b^2}{2\delta} \text{ or equivalently, } C, C' \leq \frac{\delta(A^2 + B^2)}{2}.$$



This indicates that the ranges of the parameters  $C$  and  $C'$  depend on those of  $A$  and  $B$ , and we can simply specify that  $C, C' \in [-\delta, \frac{\delta(A^2+B^2)}{2}]$ .

We also recall that the width  $W$  of a digital plane, converted from a digital annulus, is given by

$$W = C - C' = \frac{rw}{\delta} \quad (7)$$

after the normalization of (6). This indicates that  $W$  is not constant, but depends on the radius  $r$  of the digital annulus. Therefore, our converted digital plane fitting problem is different from the standard one [6] in that the width  $W$  is not a constant. Since we obtain from (6)

$$r = \sqrt{\frac{\delta(A^2 + B^2)}{2} - 2\delta C} + \frac{w}{2}, \quad (8)$$

we can also say that  $W$  depends on  $A$ ,  $B$  and  $C$ . In other words, once we set the values for  $A$ ,  $B$  and  $C$ , we can automatically set the value for  $r$ , and thus for  $W$  as well.

For this converted digital plane fitting problem, we apply the approximate method, which is originally proposed for the standard digital plane fitting problem [6]. We show in the next section that the dependency of  $W$  on  $A$ ,  $B$  and  $C$  does not critically obstruct applying the approximate method to our converted digital plane fitting problem.

### 3 Approximate Method for Digital Annulus Fitting

#### 3.1 Method Outline

The proposed digital annulus fitting algorithm consists of four steps, as described in Algorithm 1. In the first step, we convert the problem of digital annulus fitting into a problem of digital plane fitting by using the notions presented in the previous section. In other words, a set of points  $S$  in the 2D grid of size  $[0, \delta]^2$  is transformed into the set of points  $S' = \{(x, y, \frac{x^2+y^2}{2\delta}) : (x, y) \in S\}$  in the 3D grid of size  $[0, \delta]^3$ . In the second step, we build a data structure with respect to  $S'$  for the approximate halfspace range searching algorithm. More precisely, we specify a query set  $\mathbb{H}$  for approximate halfspace range searching with a given bounded error  $\varepsilon$ , and generate a data structure that allows to answer the following query: for any plane  $h \in \mathbb{H}$ , how many points of  $S'$  are on or below  $h$  with the bounded error  $\varepsilon$ . In the third step, we first modify the query for the digital plane fitting with width  $W + 5\varepsilon$ , *i.e.*, we count the number of points of  $S'$  that lie between two parallel planes with distance  $W + 5\varepsilon$ . We then find an approximate digital plane with the maximum number of inliers. The final step is to interpret the result of the converted problem as that of the original digital annulus problem.

As Steps 1 and 4 are already explained in the previous section, we describe Steps 2 and 3. Furthermore, we discuss the complexity of our proposed method.

---

**Algorithm 1.** Approximate digital annulus fitting

---

**Input:** A set  $S$  of discrete points bounded in  $[0, \delta]^2$  grid, an approximation error  $\varepsilon > 0$ , a digital annulus width  $w > 0$ .

**Output:** The fitted digital annulus  $\mathbf{A}$  containing the maximum number of inliers in  $S$ .

- 1 Convert Problem 1 into Problem 2;
  - 2 Build a data structure for approximate halfspace range searching;
  - 3 Make queries about range counting for all of the approximate digital planes and Find the one that contains the maximum number of inliers;
  - 4 Interpret the result of the digital plane fitting as digital annulus fitting.
- 

### 3.2 Data Structure for Approximate Halfspace Range Searching

As this part is based on the work of Fonseca *et al.* [9], we first summarize their method, and then explain the changes suitable for our problem.

Given a set of 3D points  $T$  and a plane  $h$  in the form  $Ax + By + z + C = 0$ , halfspace range searching (or counting) is the problem of counting the number of points of  $T$  that are on, or below  $h$ . In order to solve this problem, Fonseca *et al.* [9] have proposed an approximate method with a bounded error  $\varepsilon > 0$ .

One of the main ideas of the method is to define a sufficiently large finite set of planes  $\mathbb{H}$ , so that any query plane is approximated by some plane in  $\mathbb{H}$  with bounded error  $\varepsilon$ . In this paper,  $\mathbb{H}$  is obtained by generating different slopes  $A, B \in [-1, 0]$  and different intercepts  $C \in [-\delta, \frac{\delta(A^2+B^2)}{2}]$ ; for their ranges, see the previous section. For  $A$  and  $B$ , the interval of their variations should be set to be  $\frac{\varepsilon}{\delta}$ , while it should be  $\varepsilon$  for  $C$  (see [9] for more details).

For each plane  $h \in \mathbb{H}$ , we compute a counting function  $f(T \cap h)$  that returns approximately the number of points of  $T$  that are on, or below  $h$ . The objective of constructing a data structure is to compute this function  $f(T \cap h)$  for all the planes  $h \in \mathbb{H}$ . Points within distance  $\varepsilon$  from a plane  $h$  may or may not be reported.

The computational complexity is described as follows:

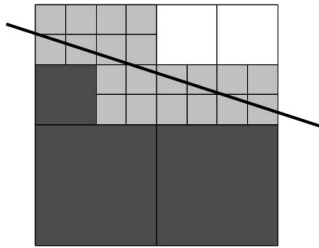
**Theorem 1.** (Fonseca *et al.* [9]) *For a set of  $N$  points in the  $[0, 1]^3$  unit grid and some  $\varepsilon > 0$ , one can build a data structure with  $O(\varepsilon^{-3})$  storage space, in  $O(N + \varepsilon^{-3} \log^{O(1)}(\varepsilon^{-1}))$  time, such that for a given query plane  $h$ , the number of points on or below  $h$  can be approximately reported in  $O(1)$  time, in the following sense: all the points (below  $h$ ) that have a larger distance than  $\varepsilon$  from  $h$  are counted. Points that are closer to  $h$  on both sides may or may not be reported.*

In fact, in order to obtain this complexity (linear to  $N$ ), we build a data structure based on an octree. Let us call a  $[0, \delta]^3$  grid a *primitive cube*, which is divided into 8 children. Its children are also divided recursively into 8 subcubes, until the size of divided cubes equals  $\varepsilon$ . Let  $X$  be a cube generated from a *primitive cube*, and  $T$  be a set of points bounded by  $X$  in dimension  $\delta$ . We then verify whether each plane of  $\mathbb{H}$  passes through  $X$ , and compute  $f(T \cap h)$ ,  $\forall h \in \mathbb{H}$  in the following way. Let  $X_i$  denote the children of  $X$ , for  $i = 1..8$ , and  $T_i = X_i \cap T$ .

We then compute  $f(T \cap h)$  by summarizing all the results from its children  $f(T_i \cap h)$ , which is also recursively computed, as follows:

$$f(T \cap h) = \begin{cases} \sum_{i=1}^8 f(T_i \cap h) & \text{if the size of } X \text{ is larger than } \varepsilon \text{ and } h \cap X \neq \emptyset, \\ 0 & \text{if the size of } X \text{ is larger than } \varepsilon \text{ and } h \text{ is below } X, \\ |T| & \text{otherwise.} \end{cases} \tag{9}$$

Finally, the data structure of a *primitive cube* is built recursively. Figure 2 illustrates a 2D example for computing the counting function  $f(T \cap h)$  of (9).



**Fig. 2.** Illustration for recursively computing  $f(T \cap h)$  in a 2D case (a line) by dividing a primitive square (corresponding to a cube in 3D) into 4 children. The light grey cells are the squares of size  $\varepsilon$  through which a query line  $h$  goes, while the dark grey cells are the squares below  $h$ . See (9) for the definition.

### 3.3 Query for Approximate Digital Plane Fitting

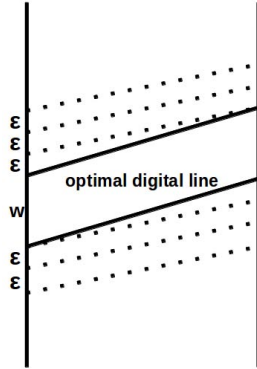
After building a data structure for a set of points  $S'$  in the  $[0, \delta]^3$  grid, we count the number of points lying in every digital plane, *i.e.*, between a pair of parallel planes with distance  $W$ . In order to calculate it approximately, we need a finite query set  $\mathbb{Q}$  of approximate digital planes for our converted digital plane fitting as follows. For each plane  $h \in \mathbb{H}$ , we take the values of  $A$ ,  $B$  and  $C$  of  $h$  for the digital plane parameters  $A$ ,  $B$ , and  $C$ , respectively. Once the values of  $A$ ,  $B$  and  $C$  are fixed, we can automatically set  $C'$  or  $W$  from (7) and (8), *i.e.*, the rest of the digital plane parameters. To generate a query for approximate digital plane fitting with a width  $W$ , we refer to the following theorem presented in [6].<sup>1</sup>

**Theorem 2.** (Aiger et al. [6]) *Given a set of  $N$  points in  $[0, \delta]^3$ , and some  $\varepsilon > 0$ ,  $W > 0$ , a digital plane of width  $W + 5\varepsilon$  that contains  $n > n_{opt}$  points, can be found in  $O(N + (\frac{\delta}{\varepsilon})^3 \log^{O(1)}(\frac{\delta}{\varepsilon}))$  time, where  $n_{opt}$  is the maximum number of points that any digital plane of width  $W$  in  $[0, \delta]^3$  can contain.*

<sup>1</sup> Even if the original theorem was proposed for  $N$  points on a  $[0, \delta]^3$  grid, the theorem itself, as well as the proposed method, is established for a more general setting such that the  $N$  points can simply be in  $[0, \delta]^3$  regardless of any grid.

In fact, we need to generate a new query set  $\mathbb{Q}$  such that any digital plane with width  $W$  is completely contained in at least one of the digital planes in  $\mathbb{Q}$ . Theorem 2 indicates that such query digital planes must have a width of at least  $W + 5\varepsilon$ . In other words, due to this setting, the important property,  $n \geq n_{opt}$ , is guaranteed. Even if  $W$  is not constant as in this paper, because it depends on each parameter set of  $A$ ,  $B$  and  $C$ , this result is still valid.

An approximate solution of the converted digital plane fitting is obtained by finding the digital plane in  $\mathbb{Q}$  in which the number of points is maximized.



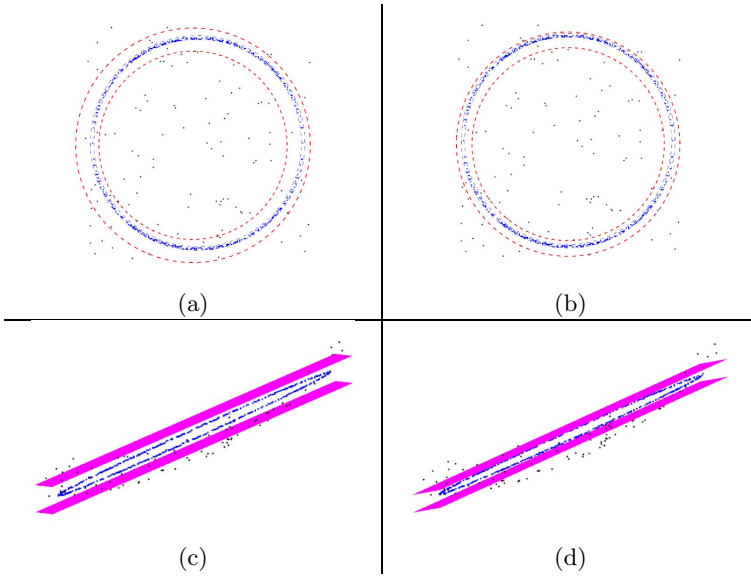
**Fig. 3.** Relationship between approximate (dotted line) and optimal digital line (continuous line). There is at least one approximate digital line that contains the optimal digital line.

### 3.4 Complexity of the Approximate Digital Annulus Fitting

As a result, we obtain the next corollary from Theorem 2 after minor adaptations.

**Corollary 1.** *Given a set of  $N$  points on a grid  $[0, \delta]^2$ , and some  $\varepsilon > 0$ ,  $w > 0$ , a digital annulus of width  $w + 5\frac{\delta}{r}\varepsilon$  that contains  $n > n_{opt}$  points, can be found in  $O(N + (\frac{\delta}{\varepsilon})^3 \log^{O(1)}(\frac{\delta}{\varepsilon}))$  time, where  $r$  is the radius of a digital annulus of width  $w + 5\frac{\delta}{r}\varepsilon$  and  $n_{opt}$  is the maximum number of points that any digital annulus of width  $w$  in  $[0, \delta]^2$  can contain.*

This corollary shows that the proposed method has linear computational complexity with respect to  $N$ . Obviously, if we decrease the bounded error  $\varepsilon$ , the computational time increases due to the second term, just like for approximated digital plane fitting; see Theorem 2. The difference to Theorem 2 is that  $\varepsilon$  cannot be tuned directly in the original 2D grid for digital annulus fitting, but in the 3D grid for the converted digital plane fitting. In other words, we can give  $\varepsilon$  as a constant ambiguous zone around the border of a digital plane in 3D (*i.e.* a zone in which points may or may not be counted), but once we project it to



**Fig. 4.** The fitted digital annuli (colored in red) for  $\varepsilon = 1.5$  (a) and  $\varepsilon = 1.0$  (b). They contain the true digital annulus (colored in blue). The converted digital planes are also illustrated for  $\varepsilon = 1.5$  (c) and  $\varepsilon = 1.0$  (d).

2D before the conversion, the corresponding ambiguous zone becomes  $\frac{\delta}{r}\varepsilon$  which is obtained from (7).

## 4 Experiments

The 2D point cloud data used for the experiments were created as follows. By using the digital annulus  $\mathbf{A}$  of (2) with  $w = 3.0$ ,  $a = b = 100.0$ , we randomly generated 400 inliers that satisfy  $\mathbf{A}$ , and  $100n$  outliers that do not satisfy  $\mathbf{A}$ , for  $n = 0, \dots, 10$ ; there are 11 variations for the number of outliers. All of the generated points are bounded in the  $[0, 200]^2$  grid. The values for the parameters of the converted digital planes  $\mathbf{P}$  of (5) are also computed:  $A = B = -0.5$ ,  $C = 30.1975$  and  $C' = 28.84$ . Note that the experiments were run on a standard PC with core i3 CPU at 2.20 GHz.

In the first experiment, we had  $n = 1$  constant (*i.e.*, 100 outliers) for the input data and observed the fitting results by using different bounded errors such that  $\varepsilon = 3.0, 2.5, 2.0, 1.5, 1.0$ . As seen in Table 1 and Figure 4, the number of inliers decreases as  $\varepsilon$  becomes smaller, and it tends to converge to the ground-truth solution. Clearly, the smaller the value of  $\varepsilon$ , the more precise the solution. Concerning the runtime, it is in fact polynomial in factor  $\varepsilon$  (see Figure 6). Therefore, we need to select an appropriate  $\varepsilon$  (for example 1.5 or 1.0 for this experiment)

**Table 1.** Results of digital annulus fitting with varied  $\varepsilon$  to the data with  $n = 1$

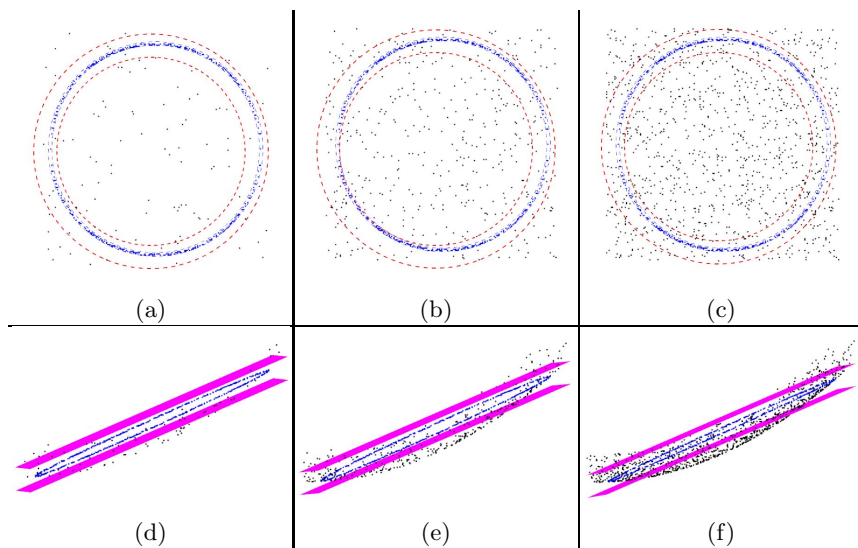
$\varepsilon$	runtime (approx. method)	parameters							
		$a$	$b$	$r$	$A$	$B$	$C$	$C'$	inliers
3.0	7.80 sec	96.0	96.0	89.62	-0.48	-0.48	34.0	18.0	441
2.5	13.11 sec	92.5	97.5	90.89	-0.46	-0.49	31.5	17.5	435
2.0	26.43 sec	96.0	104.0	95.03	-0.48	-0.52	33.0	22.0	425
1.5	59.46 sec	99.0	96.0	90.64	-0.48	-0.50	31.5	22.5	419
1.0	200.54 sec	100.0	98.0	89.46	-0.50	-0.49	32.0	26.0	406
ground truth		100.0	100.0	90.5	-0.50	-0.50	30.20	28.84	400

for assuring a reasonable runtime and an approximate solution which is not far from the ground-truth.

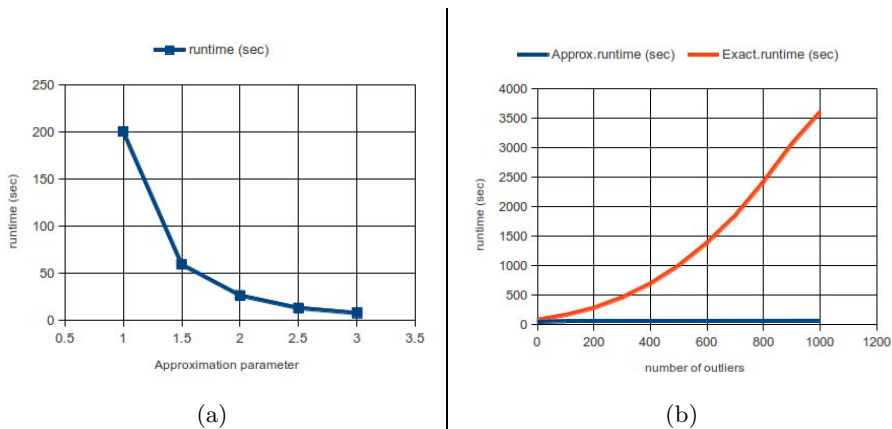
In the second experiment, we decided to use  $\varepsilon = 1.5$  for digital annulus fitting, for all the input data sets (*i.e.*, different numbers of outliers,  $100n$  for  $n = 0, \dots, 10$ ). As seen in Table 2 and Figure 5, numbers of inliers are always greater than 400 (*i.e.*, the number of inliers for the ground-truth digital annulus) and the parameter values are almost similar, while the runtimes are linear - almost constant (around 1 minute). For comparison, the exact algorithm [8] with computational complexity in  $O(N^3 \log N)$  was also applied. The obtained parameter values are  $a = 99.25$ ,  $b = 99.5$  and  $r = 90.65$  for all  $n$ . The runtimes, shown in Table 2 and Figure 6, are indeed polynomial with respect to the number of points.

**Table 2.** Results of digital annulus fitting with  $\varepsilon = 1.5$  to the data with  $100n$  outliers. The runtimes of the approximation method and the exact method are compared.

outliers ( $100n$ )	runtime (approx. method)	parameters								runtime (exact. method)
		$a$	$b$	$r$	$A$	$B$	$C$	$C'$	inliers	
0	46.20 sec	96.0	99.0	90.65	-0.48	-0.495	31.5	22.5	400	1 m 22 sec
100	59.46 sec	96.0	99.0	90.65	-0.48	-0.495	31.5	22.5	419	2 m 45 sec
200	61.48 sec	94.5	96.0	92.46	-0.47	-0.48	28.5	19.5	437	4 m 45 sec
300	63.83 sec	94.5	96.0	92.45	-0.47	-0.48	28.5	19.5	454	7 m 43 sec
400	63.94 sec	94.5	96.0	92.45	-0.47	-0.48	28.5	19.5	476	11 m 37 sec
500	64.57 sec	94.5	96.0	92.45	-0.47	-0.48	28.5	19.5	500	16 m 43 sec
600	65.00 sec	96.0	99.0	90.65	-0.48	-0.495	31.5	22.5	519	23 m 8 sec
700	65.22 sec	96.0	102.0	90.66	-0.48	-0.51	33.0	24.0	551	30 m 49 sec
800	65.77 sec	96.0	99.0	90.65	-0.48	-0.495	31.5	22.5	570	40 m 24 sec
900	66.04 sec	96.0	99.0	90.65	-0.48	-0.495	31.5	22.5	591	51 m 3 sec
1000	66.18 sec	96.0	99.0	90.65	-0.48	-0.495	31.5	22.5	618	64 m 8 sec
ground truth		100.0	100.0	90.5	-0.50	-0.50	30.20	28.84	400	



**Fig. 5.** The fitted digital annuli (colored in red) with  $100n$  outliers in input for  $n = 1$  (a),  $n = 4$  (b),  $n = 10$  (c). They contains the true digital annulus (colored in blue). The converted digital planes are also illustrated for  $n = 1$  (d),  $n = 4$  (e),  $n = 10$  (f).



**Fig. 6.** Runtimes of digital annulus fitting with respect to varied  $\varepsilon$  (a) and to number of outliers (b)

### 5 Conclusion

This paper presented an approximation method for fitting a digital annulus. Given  $N$  points in a  $[0, \delta]^2$  grid, a width  $w$  and an approximation parameter  $\varepsilon$ , the fitted digital annulus that contains the maximum number of points with a

width  $w + 5\frac{\delta}{r}\varepsilon$ , is found in  $O(N + \varepsilon^{-3}\log^{O(1)}(\varepsilon^{-1}))$  time. The method is linear in the number of points  $N$ , but it is polynomial in approximation parameter  $\varepsilon$ . Therefore, we need to find an appropriate  $\varepsilon$  for assuring a reasonable runtime. The method is also robust to outliers. The number of inliers converges to the optimal solution when  $\varepsilon$  decreases.

## References

1. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge University Press (2004)
2. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical recipes: the art of scientific computing, 3rd edn. Cambridge University Press (2007)
3. Rousseeuw, P.J.: Least median of squares regression. Journal of the American Statistical Association 79(388), 871–880 (1984)
4. 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)
5. Zrour, 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 21, 45–47 (2011)
6. Aiger, D., Kenmochi, Y., Talbot, H., Buzer, L.: Efficient Robust Digital Hyperplane Fitting with Bounded Error. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) DGCI 2011. LNCS, vol. 6607, pp. 223–234. Springer, Heidelberg (2011)
7. Andres, E., Jacob, M.-A.: The Discrete Analytical Hyperspheres. IEEE TVCG 3(1), 75–86 (1997)
8. Largeteau-Skapin, G., Zrour, R., Andres, E.:  $O(n^3 \log n)$  Time Complexity for the Optimal Consensus Set Computation for 4-Connected Digital Circles. In: Gonzalez-Diaz, R., Jimenez, M.-J., Medrano, B. (eds.) DGCI 2013. LNCS, vol. 7749, pp. 241–252. Springer, Heidelberg (2013)
9. da Fonseca, G.D., Mount, D.M.: Approximate range searching: The absolute model. Comput. Geom. 43, 434–444 (2010)
10. O'Rourke, J., Kosaraju, S.R., Megiddo, N.: Computing circular separability. Discrete and Computational Geometry 1(1), 105–113 (1986)
11. Andres, E., Acharya, R., Sibata, C.: Discrete Analytical Hyperplanes. GMIP 59(5), 302–309 (1997)
12. Provot, L., Gérard, Y.: Estimation of the Derivatives of a Digital Function with a Convergent Bounded Error. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) DGCI 2011. LNCS, vol. 6607, pp. 284–295. Springer, Heidelberg (2011)
13. Fiorio, C., Toutant, J.-L.: Arithmetic Discrete Hyperspheres and Separatingness. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 425–436. Springer, Heidelberg (2006)
14. Costa, L.F.D., Cesar, R.M.: Shape analysis and classification: Theory and practice, 1st edition. CRC Press (2000)
15. Ding, D., Ping, X., Hu, M., Wang, D.: Range image segmentation based on randomized Hough transform. Pattern Recognition Letters 26(13), 2033–2041 (2005)



# Arc Recognition on Irregular Isothetic Grids and Its Application to Reconstruction of Noisy Digital Contours

Jean-Luc Toutant<sup>1,2</sup>, Antoine Vacavant<sup>1,2</sup>, and Bertrand Kerautret<sup>3</sup>

<sup>1</sup> Clermont Université, Université d'Auvergne, ISIT, BP10448, F-63000  
Clermont-Ferrand

<sup>2</sup> CNRS, UMR6284, BP10448, F-63000 Clermont-Ferrand  
{jean-luc.toutant,antoine.vacavant}@udamail.fr

<sup>3</sup> Université de Nancy, LORIA, UMR7503 CNRS, F-54506, France  
bertrand.kerautret@univ-lorraine.fr

**Abstract.** In the present paper, we introduced an arc recognition technique suitable for irregular isothetic object. It is based on the digital inter-pixel (DIP) circle model, a pixel-based representation of the Kovalévsky's circle. The adaptation to irregular image structurations allows us to apply DIP models for circle recognition in noisy digital contours. More precisely, the noise detector from Kerautret and Lachaud (2009) provides a multi-scale representation of the input contour with boxes of various size. We convert them into an irregular isothetic object and, thanks to the DIP model, reduce the recognition of arcs of circles in this object to a simple problem of point separation.

**Keywords:** Arc recognition, Irregular isothetic grid, Digital circle.

## 1 Introduction

The recognition of circles from noisy image data has been widely studied since 60's, and was first introduced by Duda and Hart [5]. The Hough Transform is a powerful tool, robust to noise and to missing parts of an object. Nevertheless, computational and storage requirements of the algorithm are  $O(n^3)$  for circles, where  $n$  is the size of the input data. Heuristic techniques [15] or non-deterministic algorithms [10] are faster but do not guarantee the same accuracy.

Arcs and circles can also be recognized from extracted contours. P. Damaschke [3] presented a linear-time algorithm by showing that the problem of circle recognition is equivalent to solve a set of inequalities in dimension 3 [11]. Online algorithms with a  $O(n^{4/3})$  time complexity are introduced in [1,16] to segment a digital curve into arcs. All of these techniques are based on the reduction of the problem to a circle separation problem. However, this kind of approaches has the main drawback of not being suitable for noisy contours.

In [12], the authors proposed a novel approach, online and linear in time, based on a tangential space. The contour needs to be first polygonized. Thus, noisy

contours can be handle by using blurred segments [4] rather than digital segment. In [13], the method is improved with the used of a noise detector [6,7] to adapt the width of each blurred segment to the data.

In the present paper, we introduce a new digital circle model (*DIP-circle*) easy to recognize throw circle separation. It extends to irregular isothetic objects and provides arc recognition in such objects. We thus propose a complete unsupervised scheme that aims at recognizing arcs of circle from noisy image data in  $O(n^{4/3})$ .

In Section 2, we introduce the definition of circle (*DIP-circle*) on which we based our work. In Section 3, we give some recalls about irregular isothetic grids and objects. We also present how *DIP-circles* extend to such grids and allow reduction of the arc recognition problem to a circle separation problem. In Section 4, we present the unsupervised arc recognition scheme and several experiments validating it and giving some clues about its accuracy.

## 2 Digital Inter-Pixel Circles

### 2.1 Basic Notions and Recalls

Let  $\{\mathbf{e}_1, \mathbf{e}_2\}$  denote the canonical basis of the Euclidean vector plane and  $\mathbf{o}$  its origin. A point  $\mathbf{x}$  is then defined by  $\mathbf{x} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2$ . Let  $\oplus$  be the Minkowski addition such that  $\mathcal{A} \oplus \mathcal{B} = \cup_{\mathbf{b} \in \mathcal{B}} \{\mathbf{a} + \mathbf{b} : \mathbf{a} \in \mathcal{A}\}$ . It is also known as dilation. The pixel associated to an integer point  $\mathbf{x}$  is the dilation  $\mathcal{P}(\mathbf{x}) = \{\mathbf{x}\} \oplus \mathcal{B}_\infty(1/2)$  where  $\mathcal{B}_\infty(1/2)$  is the ball of radius  $1/2$  based on the  $\ell^\infty$ -norm, or, in other words, the square of side 1 centered at  $\mathbf{x}$ . The *interface* between two pixels is their intersection. A *digital object* is a set of integer points or a set of pixels, depending of the context.

Two integer points  $\mathbf{x}$  and  $\mathbf{y}$  are said to be *4-adjacent* if  $\|\mathbf{x} - \mathbf{y}\|_1 = |x_1 - y_1| + |x_2 - y_2| = 1$ . Similarly, two integer points  $\mathbf{x}$  and  $\mathbf{y}$  are said to be *8-adjacent* if  $\|\mathbf{x} - \mathbf{y}\|_\infty = \max\{|x_1 - y_1|, |x_2 - y_2|\} = 1$ . By extension, two pixels are *k-adjacent* ( $k = 4$  or  $k = 8$ ) if their associated integer points are *k-adjacent*. We also called them *k-neighbors*.

A *k-path* is a sequence of integer points such that every two consecutive points, in the sequence, are *k-adjacent*. If each point appears only one time in the path and admits at most two *k-neighbors* in, then it is called a *k-arc*. A digital object  $E$  is *k-connected* if there exists a *k-path* in  $E$  between any two points of  $E$ . A maximum *k-connected* subset of  $E$  is called a *k-connected component*. Let us suppose that the complement of a digital object  $E$ ,  $\mathbb{Z}^n \setminus E$  admits exactly two *k-connected* components  $F_1$  and  $F_2$ , or, in other words, that there exists no *k-path* joining integer points of  $F_1$  and  $F_2$ , then  $E$  is said to be *k-separating* in  $\mathbb{Z}^n$ .

### 2.2 A New Digital Circle Model

Let  $c_{\mathbf{c},r}$  be the characteristic polynomial of a circle of center  $\mathbf{c} \in \mathbb{R}^2$  and  $r \in \mathbb{R}_+$ . One has:

$$\forall \mathbf{x} \in \mathbb{R}^2 : c_{\mathbf{c},r}(\mathbf{x}) = (x_1 - c_1)^2 + (x_2 - c_2)^2 - r^2.$$

**Definition 1 (Digital Inter-pixel Circle).** Let  $\mathbf{c} \in \mathbb{R}^2$  and  $r > \sqrt{2}/2$ . The digital inter-pixel circle (*DIP-circle for short*),  $C_{\text{DIP}}(\mathbf{c}, r)$  is defined as follows:

$$C_{\text{DIP}}(\mathbf{c}, r) = \left\{ \mathbf{p} \in \mathbb{Z}^2 : \begin{array}{l} -|p_1 - c_1| - |p_2 - c_2| - \frac{1}{2} < c_{\mathbf{c},r}(\mathbf{p}) \\ \text{and} \\ c_{\mathbf{c},r}(\mathbf{p}) \leq |p_1 - c_1| + |p_2 - c_2| - \frac{1}{2} \end{array} \right\}.$$

*Property 1 (Topology of a DIP-circle).* A *DIP-circle* is a 4-connected and 8-separating set.

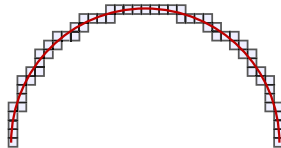
Let  $\mathbf{c} \in \mathbb{R}^2$  and  $r \in \mathbb{R}$ . Let  $D$  be the set of pixels with associated integer point in the disk of center  $\mathbf{c}$  and radius  $r$ . The Kovalevsky’s circle [9] of center  $\mathbf{c}$  and radius  $r$  is then the set of vertices of the inter-pixel boundary of  $D$  (the interfaces of the pixels in  $D$  with pixels not in). The *DIP-circle* model is strongly related to this type of circles.

*Property 2 (DIP-circle and Kovalevsky’s circle).* The Kovalevsky’s circle of center  $\mathbf{c}$  and radius  $r$  defines the same set of points as  $C_{\text{DIP}}(\mathbf{c}', r) \oplus (-1/2, -1/2)$  with  $\mathbf{c}' = \mathbf{c} \oplus (1/2, 1/2)$ .

Kovalevsky’s circles and *DIP-circles* are identical up to a translation. Kovalesky’s circles thus allow us to link *DIP-circles* with the inter-pixel boundary.

### 2.3 DIP-Circles Recognition

Circle recognition from inter-pixel boundary (vertices and edges) directly reduces to a circle separation problem: inner and outer points are easily obtained with boundary tracking. From *DIP-circles* - only the vertices of the inter-pixel boundary - the reduction to a circle separation problem is no more trivial. Sometimes *DIP-circles* contain pixels with more than two 4-neighbors and we lack information to track the boundary (one neighbor is the previous pixel in the tracking and one can not decide which of the two remaining pixels is the next one...). Thus, we only consider recognition process of *DIP-circle* performed on 4-arcs.



**Fig. 1.** In a 4-arc, a *DIP-circle* may be recognized by resolving a separation problem between the extremities of each interface between two 4-adjacent pixels

*Property 3 (Characterization of 4-arcs of DIP-circles).* Let  $\mathbf{p}$  and  $\mathbf{q}$  be two 4-adjacent integer points of  $\mathbb{C}_{\text{DIP}}(\mathbf{c}, r)$  such that they form a 4-arc with their 4-neighbors in  $\mathbb{C}_{\text{DIP}}(\mathbf{c}, r)$ . Then, we have:

- both  $\mathcal{P}(\mathbf{p})$  and  $\mathcal{P}(\mathbf{q})$  deprived of their interface intersect the circle  $\mathcal{C}(\mathbf{c}, r)$ ,
- the circle  $\mathcal{C}(\mathbf{c}, r)$  intersects only in one point the interface between  $\mathcal{P}(\mathbf{p})$  and  $\mathcal{P}(\mathbf{q})$  deprived of its end points at a distance to  $\mathbf{c}$  lower than  $r$ .

Property 3 allows to easily convert the recognition of arcs of *DIP*-circle into a circle separation problem: the parameters of the recognized *DIP*-circle are the same of those of a circle which separates the both extremities of each interface between two 4-adjacent pixels in a 4-arc, as in Figure 1.

### 3 Digital Inter-Pixel Circles on $\mathbb{I}$ -grids

We first recall the  $\mathbb{I}$ -grid (Irregular Isothetic grid) model [2,17]:

**Definition 2 (2-D  $\mathbb{I}$ -grid).** Let  $\mathcal{R}$  be a closed rectangular subset of  $\mathbb{R}^2$ . A 2-D  $\mathbb{I}$ -grid  $G$  is a tiling of  $\mathcal{R}$  with closed rectangular cells whose edges are parallel to the  $X$  and  $Y$  axes, and whose interiors have a pairwise empty intersection. The position of each cell  $R$  is given by its center point  $(x_R, y_R) \in \mathbb{R}^2$  and its length along  $X$  and  $Y$  axes by  $(l_R^x, l_R^y) \in \mathbb{R}_+^{*2}$ .

This model permits to generalize many irregular image representations such as quadtrees,  $kd$ -trees, run-length encodings, and the geometry of frames encoded with video coding standards like MPEG, H.264, etc.

We also define some topological objects on  $\mathbb{I}$ -grids, thanks to the following definitions.

**Definition 3 (ve-adjacency and e-adjacency).** Let  $R_1$  and  $R_2$  be two cells.  $R_1$  and  $R_2$  are *ve-adjacent* (vertex and edge adjacent) if :

$$\text{or } \begin{cases} |x_{R_1} - x_{R_2}| = \frac{l_{R_1}^x + l_{R_2}^x}{2} \text{ and } |y_{R_1} - y_{R_2}| \leq \frac{l_{R_1}^y + l_{R_2}^y}{2} \\ |y_{R_1} - y_{R_2}| = \frac{l_{R_1}^y + l_{R_2}^y}{2} \text{ and } |x_{R_1} - x_{R_2}| \leq \frac{l_{R_1}^x + l_{R_2}^x}{2} \end{cases}$$

$R_1$  and  $R_2$  are *e-adjacent* (edge adjacent) if we consider an exclusive “or” and strict inequalities in the above *ve-adjacency* definition. The letter  $k$  may be interpreted as *e* or *ve* in the following definitions.

Another important notion for our work is the concept of order on  $\mathbb{I}$ -grids. In the following, we will consider the total order relations based on the cell borders. We denote the left, right, top and bottom borders of a cell  $R$  respectively  $R^L$ ,  $R^R$ ,  $R^T$  and  $R^B$ . The abscissa of  $R^L$ , for example, is equal to  $x_R - (l_R^x/2)$ . In the following, we also denote by  $\leq_{\bar{x}}$  (resp.  $\leq_{\bar{y}}$ ) the natural order relation along  $X$  (resp.  $Y$ ) axis. It is legitimate to use the order  $\leq_{\bar{x}}$  on left and right borders of cells and the order  $\leq_{\bar{y}}$  on top and bottom borders of cells.

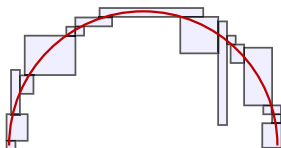
**Definition 4 (Order relations on an  $\mathbb{I}$ -grid).** Let  $R_1$  and  $R_2$  be two cells of an  $\mathbb{I}$ -grid  $G$ . We define the total order relations  $\preceq^L$  and  $\preceq^T$ , based on the cells borders:

$$R_1 \preceq^L R_2 \Leftrightarrow R_1^L \underset{x}{<} R_2^L \vee \left( R_1^L = R_2^L \wedge R_1^T \underset{y}{<} R_2^T \right);$$

$$R_1 \preceq^T R_2 \Leftrightarrow R_1^T \underset{y}{<} R_2^T \vee \left( R_1^T = R_2^T \wedge R_1^L \underset{x}{<} R_2^L \right).$$

A  $k$ -path from  $R$  to  $R'$  is a sequence of cells  $(R_i)_{1 \leq i \leq n}$  with  $R = R_1$  and  $R' = R_n$  such that for any  $i$ ,  $2 \leq i < n$ ,  $R_i$  is  $k$ -adjacent to  $R_{i-1}$  and  $R_{i+1}$ .

**Definition 5 ( $k$ -arc).** Let  $A = (R_i)_{1 \leq i \leq n}$  be a  $k$ -path from  $R_1$  to  $R_n$ . Then  $A$  is a  $k$ -arc iff each cell  $R_i$  has exactly two  $k$ -adjacent cells in  $A$  except  $R_1$  and  $R_n$  which have only one  $k$ -adjacent cell in  $A$ . The cells  $R_1$  and  $R_n$  are called the extremities of  $A$ .



**Fig. 2.** In an  $e$ -arc on  $\mathbb{I}$ -grids, a  $DIP$ -circle may be recognized by resolving a separation problem between the extremities of each interface between two  $e$ -adjacent cells

Our purpose is now to extend the  $DIP$ -circle model we have previously presented to this kind of grids. The extension we use is natural.

*Property 4 (Characterization of  $e$ -arcs of  $DIP$ -circles).* Let  $R_1$  and  $R_2$  be two  $e$ -adjacent cells of  $C_{DIP}(\mathbf{c}, r)$  such that they form a  $e$ -arc with their  $e$ -neighbors in  $C_{DIP}(\mathbf{c}, r)$ . Then, we have:

- both  $R_1$  and  $R_2$  deprived of their interface intersect the circle  $\mathcal{C}(\mathbf{c}, r)$ ,
- the circle  $\mathcal{C}(\mathbf{c}, r)$  intersects only in one point the interface between  $R_1$  and  $R_1$  deprived of its end points at a distance to  $\mathbf{c}$  lower than  $r$ .

The reduction of the arc recognition problem in an  $\mathbb{I}$ -grid to a circle separation problem follows the same principle as in a regular grid: the parameters of the recognized  $DIP$ -circle are the same of those of a circle which separates the both extremities of each interface between two  $e$ -adjacent cells of the initial  $e$ -arc, as in Figure 2.

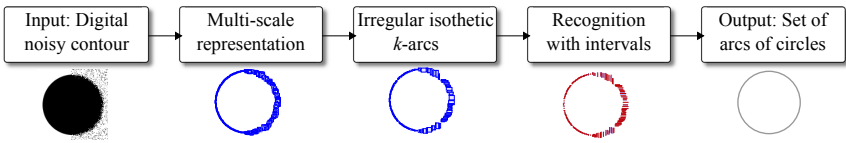
## 4 Application to Noisy Contour Reconstruction

In this section, we attempt to build a global system of recognition of arcs of circles through digital contours. Our goal is to confront the concepts previously

introduced to practical cases. At this point, it is mainly used to recognize circles, since it applies only to contours of disk-shaped objects. The limitation comes from the difficulties to deduce global  $k$ -arcs from any irregular isothetic object. At the end of the present paper, we point out some perspectives on how to overcome this limitation.

#### 4.1 Method

The flowchart of our method is given in Figure 3. Our arc reconstruction method is divided into three main steps: a multi-scale noise detection, a topological reconstruction of the resulting object and a translation into a circle separation problem.

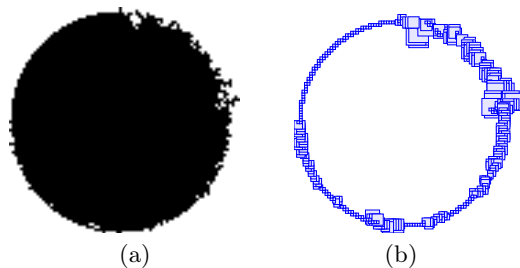


**Fig. 3.** The global scheme of our method, where an example leads to the recognition of one arc of circle (the complete circle)

The noise detector developed in [6,7] exploits the known asymptotic properties of maximal straight segments for flat parts, or smooth curved parts, of a digital contour. Asymptotic properties hold for finer and finer scale, but it appears that in practice, it fits also well for coarser and coarser scales. The contour is then subsampled at different scales and the asymptotic properties are tested. If they are not satisfied, the finer scales are removed and the properties tested again. Each contour point is then covered by a square cell, called *meaningful box*, whose size is the finest resolution determined by this scale detection. Finally, the higher the local amount of noise is, the bigger the meaningful box is. In Figure 4, we present the result of this process on a sample noisy image, the set  $\mathcal{M}$  of the meaningful boxes deduced from the contour of the black object.

As shown in Figure 4(b), the meaningful boxes of  $\mathcal{M}$  overlap and thus cannot be viewed as an irregular isothetic object directly (Definition 2). However each one contains a given number of pixels (at the initial resolution) so that the set of boxes covers a subset of the input image. This subset  $\mathcal{P}$ , which is an irregular isothetic object, is transformed into four  $k$ -arcs (Definition 5). Instead of using a unique order relation over the complete object  $\mathcal{M}$ , we use either  $\preceq^L$  or  $\preceq^T$ , depending on the position of the treated cells. More precisely, we use the following procedure (see also Figure 5):

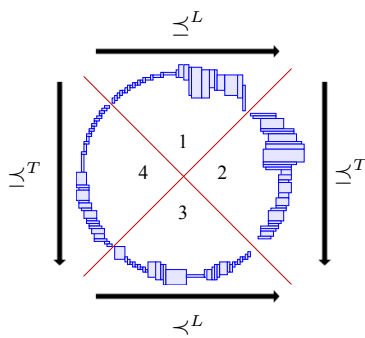
- (1) We compute the barycenter  $\bar{\mathbf{p}}$  of the input digital object, based on the center of the pixels belonging to  $\mathcal{P}$ .
- (2) We construct two straight lines  $\ell_1$  and  $\ell_2$  passing through  $\bar{\mathbf{p}}$ , with slopes 1 and -1. These lines define four quadrants in the Euclidean plane.



**Fig. 4.** We analyze the noise of a digital contour to get a multi-scale structure

(3) We use the topological reconstruction algorithm from [17] with  $\preceq^L$  order for pixels of  $\mathcal{P}$  in quadrants 1 and 3 and  $\preceq^T$  order for pixels of  $\mathcal{P}$  in quadrants 2 and 4.

The result is a set of four  $e$ -arcs, one in each quadrant defined by  $\ell_1$  and  $\ell_2$ . Cells on both side of the boundary between two quadrant can overlap. For clarity issues, overlapping cells are not drawn in Figure 5.



**Fig. 5.** The construction of the barycenter at the intersection of  $\ell_1$  and  $\ell_2$

Then, we consider the interface (Euclidean segments shared) between two consecutive cells in the  $e$ -arcs over the quadrants 1 to 4, in the clockwise order. Finding circles inside the original noisy contour can now be considered as a problem of separation of the points defined as the extremities of these segments. To do so, we apply Algorithm 1 introduced by Roussillon *et al.* [16]. It is online and its time complexity is basically  $\mathcal{O}(n_S^2)$  ( $n_S$  is the number of input segments), but may be optimized to  $\mathcal{O}(n_S^{4/3})$  and even seems to be linear in practice.

Algorithm 1 first aims to recognize a set of straight lines  $\mathcal{P}_l(A_j)$  passing through the input intervals (also known as *preimage*), from line 3 to 7 (Figure 6(a)). This recognition process is realized in an incremental way, by inserting input segments in the arc  $A_j$ . The end of this phase means that no straight line can pass through  $A_j$ . Then, a fixed point  $\mathbf{p}$  is selected from one of the extremities of the interval that fails the first step. The arc  $A_j$  is updated so that it represents

---

**Algorithm 1.** Computation of a set of arcs of circles from a set of segments

---

**input** : A set of  $n_S$  intervals (segments)  $\mathcal{S} = \{S_i\}_{1 \leq i \leq n_S}$   
**output**: A set of  $n_A$  arcs of circles  $\mathcal{A} = \{A_j\}_{1 \leq j \leq n_A}$

```

1  $i \leftarrow 1, j \leftarrow 1$  ;
2 do
3    $A_j \leftarrow \{S_i\}$  ; { $A_j$  is considered as a straight arc of radius  $\infty$ }
4   do
5      $i \leftarrow i + 1$  ;
6      $A_j \leftarrow A_j \cup \{S_i\}$  ;
7   while  $i \leq n_S \wedge \mathcal{P}_l(A_j) \neq \emptyset$  ;
   { $A_j$  is not straight anymore, since its preimage  $\mathcal{P}_l(A_j)$  is empty }
8   Choose a point  $\mathbf{p} \in S_i$  ; { $\mathbf{p}$  is the fixed point}
9   Update radius and support points of  $A_j$  ;
   { $A_j$  is the arc of circle passing through  $\mathbf{p}$  and two support points in  $A_j$  }
10  do
11     $i \leftarrow i + 1$  ;
12    Update radius, fixed point and support points of  $A_j$  with  $S_i$  ;
13  while  $i \leq n_S \wedge \mathcal{P}_c(A_j) \neq \emptyset$  ;
14   $\mathcal{A} \leftarrow \mathcal{A} \cup \{A_j\}$  ;
15   $j \leftarrow j + 1$  ;
16 while  $i \leq n_S$  ;

```

---

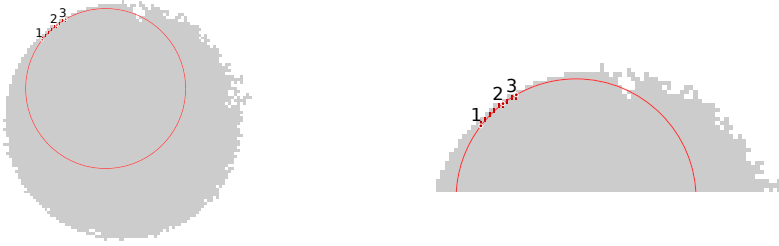
a circle passing through  $\mathbf{p}$  and two support points of  $A_j$ . These points belong to the partial upper and lower convex envelopes of the set of points to separate [16]. While intervals are added incrementally, the support points and the fixed point are updated to fit the circle at best to the input segments (Figure 6(b)). This second phase, from line 10 to 13 in Algorithm 1, finishes once the set of circles passing through  $A_j$ , denoted by  $\mathcal{P}_c(A_j)$ , is empty. The arc  $A_j$  is added to the set of recognized arcs  $\mathcal{A}$ , lines 14 and 15, and a new arc may be constructed. If the number of segments  $n_S$  is achieved at this moment, the result only contains one arc, as in Figure 6(c).

## 4.2 Experiments

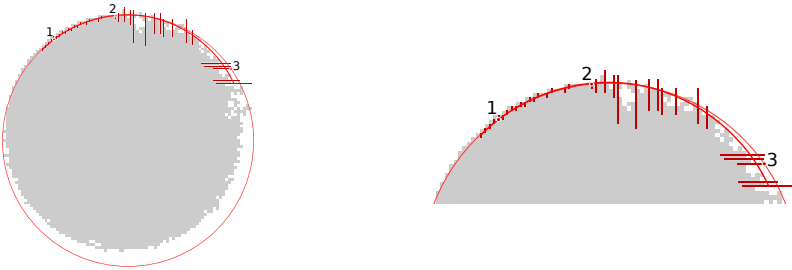
**Synthetic Image.** Throughout Section 4.1, we illustrate the different steps of our method with the example of a circle with non uniform noise distribution. It is reconstructed as a single circle with parameters different from those of the circle used to generate the image (see the result in Figure 6(c)). The circle separation problem our method solves has generally many solutions and Algorithm 1 chooses only one. We have checked that the original circle is also one of the solutions.

**Sun Image at Different Resolutions.** In Figure 7, we consider a sun image of size  $n \times n$ , with two different resolutions  $n = 512$  and  $n = 1024$ . A contour of the sun is extracted by a simple thresholding. In both cases, our method succeeds in reconstructing only a single circle.

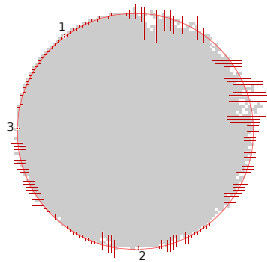




(a)



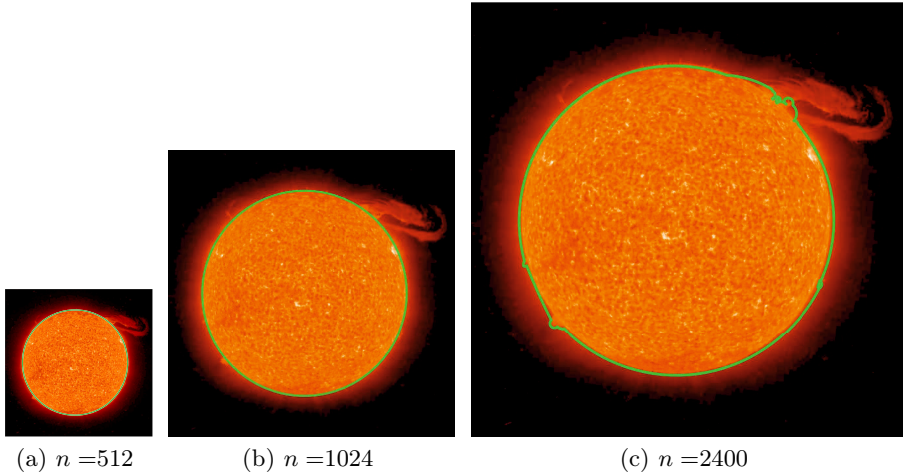
(b)



(c)

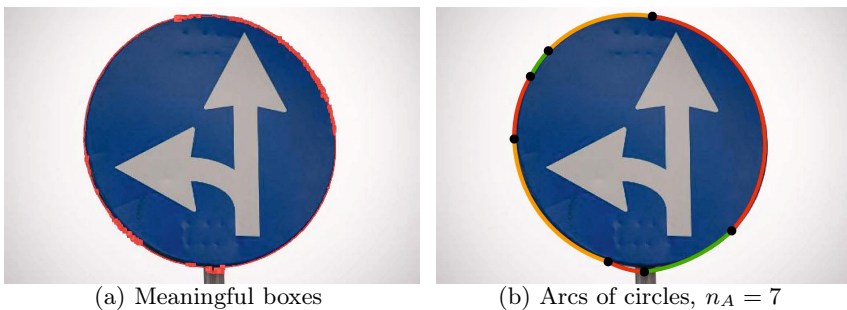
**Fig. 6.** The first step of Algorithm 1 consists in recognizing a segment (a). When this phase stops, a first arc of circle is constructed, passing through a fixed point (indexed 3), and two support points (indexed 1 and 2). This process continues, and these three points are updated, leading to the update of the circle recognized (b). The process stops when all the input intervals are treated (c).

When we consider the highest resolution available for this picture ( $n = 2400$ ), we obtain a decomposition of 33 arcs of circles. At this scale the small details of the digital contour are no more considered as noise and the algorithm is now able to detect arcs defined in finer resolution.



**Fig. 7.** The extraction of one circle realized on an image of Sun, for the 512 and 1024 resolutions. For the original resolution (2400), our method leads to a decomposition of 33 arcs of circles.

**Non-circular Shapes.** We have also tested our method on an image of road sign (of  $520 \times 340$  pixels), where the contour is roughly elliptic. In this case again, we are able to extract arcs of circles close to the real contour, as illustrated in Figure 8.



**Fig. 8.** Extraction of the meaningful boxes from the road sign (a), and its associated decomposition into 7 arcs of circles (b)

## 5 Conclusion and Future Works

In the present paper, we have introduced a new circle model close to the Kovalevsky's one. Thanks to its links with the inter-pixel boundary, it is provided with interesting properties to be recognized through the resolution of a circle separation problem. Moreover, it extends nicely to irregular isothetic grids and objects.

From a practical point of view, we have designed a simple arc recognition scheme. It involves recent results in the fields of digital geometry about noise detection, isothetic irregular objects and digital circles. Each step can be proceeded in linear time with respect to the length of the digital contour analyzed. Applications to synthetic images ensure that we are able to evaluate a promising method, whereas applications to real images show its concrete interest and efficiency. Nevertheless, deeper tests and comparisons with existing recognition schemes should be performed.

In future works, we plan to improve the decomposition of the irregular isothetic object into  $e$ -arcs. In the scheme presented in the present paper, we can only process disk-shaped objects. This is due to the naive decomposition of the isothetic object used to reconstruct  $e$ -arcs : we break it, according to the case of circles, into quadrants where the total order relation  $\preceq^L$  applies and quadrants where the total order relation  $\preceq^T$  applies. This limitation can be overcome by adapting the decomposition to the treated object. A first approach could consist in recognizing segments in the set of the meaning boxes; their directions are indeed sufficient to choose the appropriate order relation for each part of the contour. A second approach could consist in computing the isothetic objects associated to each total order relation. Then, an appropriate decomposition is a cover of the set of meaningful boxes by  $e$ -arcs of these isothetic objects.

The decomposition in both arcs and segments of noisy contours is also a question we intend to address. Indeed, tools and approaches for both primitives are similar [18,19].

## References

1. Coeurjolly, D., Gerard, Y., Reveillès, J.P., Tougne, L.: An elementary algorithm for digital arc segmentation. *Dis. Applied Maths* 139(1-3), 31–50 (2004)
2. Coeurjolly, D., Vacavant, A.: Separable Distance Transformation and its Applications. In: Brimkov, V., Barneva, R. (eds.) *Digital Geometry Algorithms. Theoretical Foundations and Applications to Computational Imaging*. Springer (2012)
3. Damaschke, P.: The linear time recognition of digital arcs. *Pat. Rec. Letters* 16(5), 543–548 (1995)
4. Debled-Rennesson, I., Feschet, F., Rouyer-Degli, J.: Optimal blurred segments decomposition of noisy shapes in linear time. *Computers & Graphics* 30(1), 30–36 (2006)
5. Duda, R.O., Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. *Com. of ACM* 15(1), 11–15 (1972)

6. 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)
7. Kerautret, B., Lachaud, J.: Meaningful scales detection along digital contours for unsupervised local noise estimation. *IEEE Pat. Analysis and Machine Intel.* 34(12), 2379–2392 (2012)
8. Kerautret, B., Lachaud, J.-O., Nguyen, T.P.: Circular Arc Reconstruction of Digital Contours with Chosen Hausdorff Error. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) DGCI 2011. LNCS, vol. 6607, pp. 247–259. Springer, Heidelberg (2011)
9. Kovalevsky, V.A.: New definition and fast recognition of digital straight segments and arcs. In: *ICPR 1990*, vol. 2, pp. 31–34 (1990)
10. Lamiroy, B., Fritz, L., Gaucher, O.: Robust Circle Detection. In: *ICDAR 2007*, pp. 526–530 (2007)
11. Megiddo, N.: Linear programming in linear time when the dimension is fixed. *Jour. of ACM* 31(1), 114–127 (1984)
12. Nguyen, T.P., Debled-Rennesson, I.: Arc Segmentation in Linear Time. In: Real, P., Diaz-Pernil, D., Molina-Abril, H., Berciano, A., Kropatsch, W. (eds.) *CAIP 2011, Part I*. LNCS, vol. 6854, pp. 84–92. Springer, Heidelberg (2011)
13. Nguyen, T.P., Kerautret, B., Debled-Rennesson, I., Lachaud, J.-O.: Unsupervised, Fast and Precise Recognition of Digital Arcs in Noisy Images. In: Bolc, L., Tadeusiewicz, R., Chmielewski, L.J., Wojciechowski, K. (eds.) *ICCVG 2010, Part I*. LNCS, vol. 6374, pp. 59–68. Springer, Heidelberg (2010)
14. Pham, S.: Digital circles with non-lattice point centers. *The Visual Computer* 9, 1–24 (1992)
15. Ayala-Ramírez, V., Garcia-Capulin, C.H., Pérez-García, A., Sánchez-Yáñez, R.E.: Circle detection on images using genetic algorithms. *Pat. Rec. Letters* 27(6), 652–657 (2006)
16. 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)
17. Vacavant, A., Coeurjolly, D., Tougne, L.: Topological and Geometrical Reconstruction of Complex Objects on Irregular Isothetic Grids. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) *DGCI 2006*. LNCS, vol. 4245, pp. 470–481. Springer, Heidelberg (2006)
18. Vacavant, A., Roussillon, T., Kerautret, B.: Unsupervised Polygonal Reconstruction of Noisy Contours by a Discrete Irregular Approach. In: Aggarwal, J.K., Barneva, R.P., Brimkov, V.E., Koroutchev, K.N., Korutcheva, E.R. (eds.) *IWCIA 2011*. LNCS, vol. 6636, pp. 398–409. Springer, Heidelberg (2011)
19. Vacavant, A., Roussillon, T., Kerautret, B., Lachaud, J.-O.: A Combined Multi-Scale/Irregular Algorithm for the Vectorization of Noisy Digital Contours. *Computer Vision and Image Understanding* (to appear, 2013)

# Reconstruction of Quantitative Properties from X-Rays\*

Fatma Abdmouleh and Mohamed Tajine

LSIIT CNRS UMR 7005, Strasbourg University  
Pôle API Boulevard Sébastien Brant 67412 Illkirch-Graffenstaden, France

**Abstract.** In some applications, the tomographic reconstruction is not an end in itself. When the goal is rather to gather information about the object being studied, the question is if it is more interesting to directly extract these information from the projections without the reconstructing step. We would then know if less projections are needed to directly get the information than to reconstruct the object. In this paper, we address the problem of extracting quantitative information about an object namely an estimation of its area, an upper and a lower bound to the perimeter given its projections from point sources.

## 1 Introduction

Tomographic reconstruction aims to reconstruct the image of an object given its projections. In some applications, this is done in the purpose of gathering information about this object. This information can be of a qualitative type: the topology (connexity, Euler number, tree of connected components), the geometry (convexity, shape). The information sought can also be of a quantitative type: perimeter, surface area, curvature, etc.

Many researches were lead to know how many projections are needed for the tomographic reconstruction and how this reconstruction is possible in an optimal way. For example, in [6] it is proven that we need three point sources to reconstruct a convex set. The aim of our study is to answer the following question: do we need less projections to directly get the information without reconstructing the image of the object? In this case, it would be more interesting to skip the reconstruction step

In literature, some papers worked on this idea. For example in [5] the smallest possible boundary length of the projected set is estimated from horizontal and vertical projections, in [3] the perimeter of convex sets is estimated from horizontal and vertical projections. In [2], decision trees are used to classify hv-convex sets only from their projections. The cited works consider projections from parallel X-rays, while we address here the problem for projections from point sources. This context is more realistic and general than the parallel X-rays

---

\* This work was supported by the Agence Nationale de la Recherche through contract ANR-2010-BLAN-0205-01.

since the latter is obtained via an approximation supposing that the point source is at an infinite distance from the object being studied.

This paper is organized as follows. In Section 2 we define the basic tools that will be used in this work. Section 3 is dedicated to the estimation of the surface area from point source projections while in Section 4 an upper and a lower bound to the perimeter are presented.

## 2 Definition and Notation

In this section we introduce the notation and define the tools that will be useful in the paper. When a statement is true as well in  $\mathbb{R}^2$  and in  $\mathbb{Z}^2$ , we use the notation  $\mathbb{S}^2$ .

All topological notion used in this paper is considered relatively to the usual topology (Euclidean topology).

With no loss of generality, we consider that the point sources are collinear on the x-axis.

We start by defining the notion of line segment in the continuous and the discrete space:

**Definition 1.** *Let  $a, b \in \mathbb{S}^2$ . We define the continuous line segment as  $[a, b] = \{\lambda a + (1 - \lambda)b \mid 0 \leq \lambda \leq 1\}$ .*

*When  $a, b \in \mathbb{Z}^2$ , we define the discrete line segment as  $\llbracket a, b \rrbracket = [a, b] \cap \mathbb{Z}^2$ .*

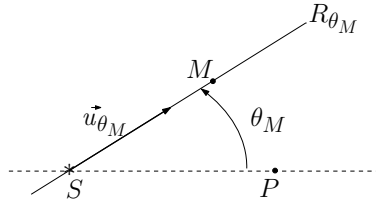
Let  $E$  be a subset of  $\mathbb{R}^2$  and  $a$  and  $b$  be two distinct points of  $\mathbb{R}^2$ . We use the following notation:

- If  $E$  is a finite set, then  $|E|$  is the cardinality of  $E$  indicating the number of elements of  $E$ .
- $\delta E$  is the boundary of  $E$ .
- $\overset{\circ}{E}$  is the topological interior of  $E$ .
- $(ab)$  is the straight line joining  $a$  and  $b$ .
- $\mathfrak{P}(E)$  is the powerset of  $E$  ( $\mathfrak{P}(E) = \{F \mid F \subseteq E\}$ ).
- A ray or a half-line  $R_\theta$  from a point  $S = (x_0, y_0)$  in the direction  $\vec{u}_\theta = (u_1, u_2)$  where  $\|\vec{u}_\theta\| = 1$ , and  $\cos \theta = u_1$  and  $\sin \theta = u_2$  can be defined in different ways:

$$\begin{aligned} R_\theta &= \{(x, y) \in \mathbb{R}^2 \mid u_2(x - x_0) - u_1(y - y_0) = 0 \text{ and } x \geq x_0\}; \\ &= \{(x_0, y_0) + \lambda \vec{u}_\theta \mid \lambda \geq 0\}; \\ &= \{M \in \mathbb{R}^2 \mid \widehat{PSM} = \theta\}; \end{aligned}$$

where  $\widehat{PSM}$  denotes the angle between  $(SP)$  and  $(SM)$  with  $P = S + (1, 0)$  (see Figure 1.). In what follows, the angle  $\widehat{PSM}$  is denoted  $\theta_M$ .

- For a point  $S \in \mathbb{R}^2$ , we define the set  $\mathcal{K}(S, \mathbb{S}^2)$  formed by all the angles of all the rays issuing from  $S$  and passing through points of  $\mathbb{S}^2$  with respect to the horizontal line passing through  $P = S + (1, 0)$ :



**Fig. 1.** The ray  $R_{\theta_M}$  is defined by the source point  $S$  and the angle  $\theta_M$

$$\mathcal{K}(S, \mathbb{S}^2) = \{ \theta_M \in [0, 2\pi[ \mid M \in \mathbb{S}^2 \} .$$

–  $\mathcal{L}(E)$  and  $\mathcal{A}(E)$  are respectively the perimeter and the surface area of  $E$ .

**Definition 2.** Consider a set  $E \subset \mathbb{R}^2$ ,  $r > 0$ .

We denote by  $r\mathbb{Z}^2$  the discrete grid having a resolution that is equal to  $p = 1/r$  called  $r$ -grid.

We define the discretization operator  $\Lambda_r : \mathfrak{P}(\mathbb{R}^2) \mapsto \mathfrak{P}(r\mathbb{Z}^2)$  such that  $\Lambda_r(E) = E_r = E \cap r\mathbb{Z}^2$ .

We now introduce the notions of projections from a point source.

The continuous projection (or  $\mathbb{R}$ -projection) of  $E \subset \mathbb{R}^2$  from the point source  $S \in \mathbb{R}^2 \setminus E$  denoted  $X_{\mathbb{R}}(E, S, \cdot) : [0, 2\pi[ \mapsto \mathbb{R}$  is:

$$X_{\mathbb{R}}(E, S, \theta) = \int_0^{+\infty} \chi_E(S + t\vec{u}_{\theta}) dt.$$

Where  $\vec{u}_{\theta} = (\cos \theta, \sin \theta)$  and

$$\chi_E(x) = \begin{cases} 1 & \text{if } x \in E \\ 0 & \text{otherwise.} \end{cases}$$

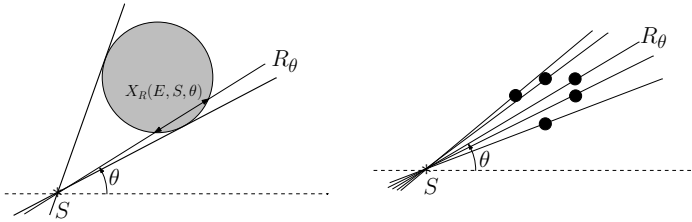
Then,  $X_{\mathbb{R}}(E, S, \theta) = \mu(E \cap R_{\theta})$  where  $\mu$  is the usual Lebesgue’s measure on  $\mathbb{R}$ .

Let  $r > 0$ , we define the notion of discrete projection from a point source on the  $r$ -grid. Let  $S \in \mathbb{R}^2$  and a finite subset  $D \subset r\mathbb{Z}^2$  such that  $S \notin D$ .

$D$  being a finite set, we have a finite number of rays issuing from  $S$  and passing through points of  $D$  and each of these rays passes through a finite number of points of  $D$ . The  $r\mathbb{Z}$ -projection of  $D$  from the point source  $S$  is the function  $X_{r\mathbb{Z}}(D, S, \cdot) : [0, 2\pi[ \mapsto \mathbb{N}$  such that:

$$X_{r\mathbb{Z}}(D, S, \theta) = |R_{\theta} \cap D| .$$

Finally, we define the support of the projection for a point source and the projected set [1].



**Fig. 2.** Continuous(left) and discrete(right) point X-rays

**Definition 3.** Let  $S \in \mathbb{S}^2$  and  $E \subset \mathbb{S}^2$  such that  $S \notin E$ . The support of  $S$ -projections of  $E$  for the point source  $S$  is the set:

$$Supp_{\mathbb{S}}(E, S) = \{ \theta \in \mathcal{K}(S, \mathbb{S}^2) \mid X_{\mathbb{S}}(E, S, \theta) \neq \emptyset \}.$$

### 3 Surface Area Estimation from Point X-Rays

In this section, we aim to find an estimation of the surface area of a set given its projections.

Consider a set  $E \subset \mathbb{R}^2$ ,  $r > 0$  and  $E_r = \Lambda_r(E)$ .

We suppose in this subsection that we have the exact  $r\mathbb{Z}$ -projections of  $E_r$  for any  $r$ . The sum of the projections of  $E_r$  is the cardinality of  $E_r$  and then is the same for any point source. Let  $n_r$  be the number of the rays from  $S$  passing through all the points of  $E_r$ . Given the projections from each of these rays, the number of the points of  $E_r$  is given by:

$$|E_r| = \sum_{j=1}^{n_r} s_j^r.$$

where  $s_j^r$  is the number of points of  $r\mathbb{Z}^2$  lying on the  $j^{th}$  ray corresponding to an angle of  $Supp_{r\mathbb{Z}}(E_r, S)$ .

In all the following, we suppose that we have the boundary  $\delta E = \Gamma_1 \cup \Gamma_2$  such that  $\Gamma_1$  and  $\Gamma_2$  respectively the graphs of continuous functions  $f_1, f_2 : [a, b] \mapsto \mathbb{R}$  with  $a, b \in \mathbb{R}$  (see Figure 3 for illustration).

For each point  $P = (p_1, p_2)$  of  $r\mathbb{Z}^2$ , we consider the pixel centered at  $P$ :  $W(P) = \{ (x, y) \in \mathbb{R}^2 \mid |x - p_1| \leq r/2; |y - p_2| \leq r/2 \}$ . The area of  $W(P)$  is then equal to  $r^2$ . This will be used to estimate the area of  $E_r$  as follows:

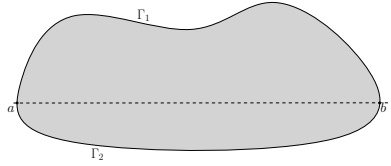
$$\mathcal{A}(E_r) = r^2 * \sum_{j=1}^{n_r} s_j^r;$$

We present in the following proposition a new estimator of the area of  $E$ :

**Proposition 1.** Given a set  $E \in \mathbb{R}^2$  with  $\delta E = \Gamma_1 \cup \Gamma_2$  such that  $\Gamma_1$  and  $\Gamma_2$  respectively the graphs of continuous functions and  $E_r = \Lambda_r(E)$  with  $r > 0$ . We have:

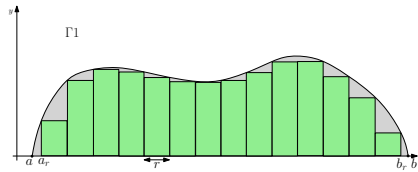
$$\lim_{r \rightarrow 0} \mathcal{A}(E_r) \rightarrow \mathcal{A}(E)$$





**Fig. 3.**  $\delta E = \Gamma_1 \cup \Gamma_2$  such that  $\Gamma_1$  and  $\Gamma_2$  are the graphs of continuous functions

*Proof.* This proof is based on the Riemann Integral theory. Indeed, consider a line  $(ab)$  that divides the boundary of  $E$  into  $\Gamma_1$  and  $\Gamma_2$  such that  $\Gamma_1$  and  $\Gamma_2$  are respectively the graphs of continuous functions  $f_1, f_2 : [a, b] \mapsto \mathbb{R}$ . We can suppose with no loss of generality that  $(ab)$  is the  $x$ -axis. We are then interested in measuring  $\mathcal{A}(E) = \int_a^b f_1(x) dx + \int_a^b f_2(x) dx$ . Let us show how to estimate  $\int_a^b f_1(x)$ . The same can be done for  $f_2$ . We will cover the considered area with rectangles of width equal to  $r$  starting from  $a_r = \lfloor \frac{a}{r} \rfloor \times r$  and ending at  $b_r = \lfloor \frac{b}{r} \rfloor \times r$  as illustrated on Figure 4.



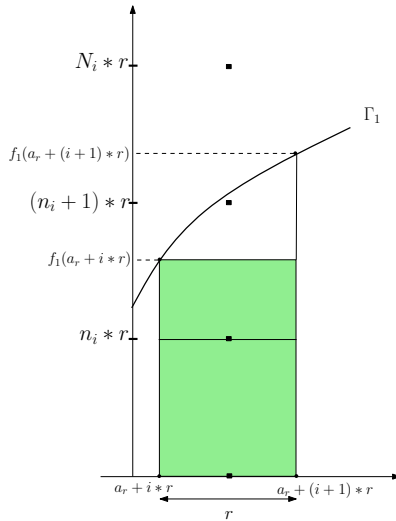
**Fig. 4.** The area of the considered set is covered by rectangles of width equal to  $r$

By the property of Riemann’s integral, we have:

$$\lim_{r \rightarrow 0} r \times \sum_i \min_{x \in [i \times r, (i+1) \times r]} f_1(a_r + x) \rightarrow \int_a^b f_1(x) dx$$

In our case, we only have the points with coordinates in  $r\mathbb{Z}^2$ . For a rectangle  $i$ , we consider  $n_i = \lfloor \frac{f_1(a_r + i \times r)}{r} \rfloor$  (see Figure 5). The additional error induced by considering  $n_i$  instead of  $f_1(a_r + i \times r)$  on each rectangle of the partition is then  $r \times (f_1(a_r + i \times r) - n_i \times r)$  where  $(f_1(a_r + i \times r) - n_i \times r) \leq r$ . Summing on all the rectangles gives then an error that is at most equal to  $(b - a) \times r$ . Yet,  $\lim_{r \rightarrow 0} (b - a) \times r \rightarrow 0$ .

There remains the parts we neglected when we started the rectangles at  $a_r$  and finished at  $b_r$ . Since  $f_1$  is continuous on the compact subset  $[a, b]$ , there exists  $M(f_1) = \max_{x \in [a, b]} (f_1(x))$ . The area of the neglected part is then at most equal to  $2 \times r \times M(f_1)$  and so it tends to 0 when  $r$  tends to 0. □



**Fig. 5.** The error on each rectangle of the partition is  $r \times (f_1(a_r + i \times r) - n_i)$

### 3.1 Example

Let  $E = [1, 2]^2$  be a square having the sides equal to 1. The projections from any point source  $S \in \mathbb{R}^2 \setminus E$  of  $E_r = A_r(E)$ , with  $r = 1/p > 0$  and  $p \in \mathbb{N}^*$ , verify the following:

$$\sum_{j=1}^{n_r} s_j^r = \left(\frac{1}{r} + 1\right)^2$$

The area of the pixel of  $r\mathbb{Z}^2$  is equal to  $r^2$ , and so :

$$\mathcal{A}(E_r) = \left(\frac{1}{r} + 1\right)^2 \times r^1 = 1 + 2r + r^2$$

Then  $\lim_{r \rightarrow 0} \mathcal{A}(E_r) \rightarrow 1 = \mathcal{A}(E)$ .

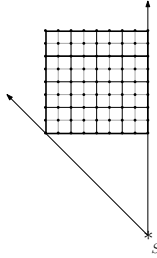
## 4 Perimeter Estimation from Point Sources

In this part we give two lower bounds and a higher bound to the perimeter of a given set from two projections.

### 4.1 Lower Bounds of the Perimeter with One Point Source

A lower bound of the perimeter of a set  $E$  is given thanks to the following property called the isoperimetric inequality [4]

$$\mathcal{L}^2(E) \geq 4\pi\mathcal{A}(E)$$



**Fig. 6.** The discretization of the square at a resolution  $1/r$  contains  $(\frac{1}{r} + 1)^2$  points of  $r - \mathbb{Z}^2$

where  $\mathcal{A}$  is the measure of the area enclosed by a curve of length  $L$ . When  $E$  is a circle, we obtain the isoperimetric equality:  $\mathcal{L}^2(E) = 4\pi\mathcal{A}(E)$ .

From the isoperimetric inequality we can then deduce the following:

**Proposition 2.** *Given  $E \subset \mathbb{R}^2$  and a point source  $S \in \mathbb{R}^2$ . The perimeter  $\mathcal{L}(E)$  of  $E$  necessarily verifies the following inequality:*

$$\mathcal{L}^2(E) \geq 4\pi \lim_{r \rightarrow 0} \mathcal{A}(E_r) \tag{1}$$

When  $E$  is convex, another lower bound of the perimeter can be given thanks to the Crofton Formula:

**Proposition 3. [Crofton Formula]** *Let  $\gamma : [0, 1] \mapsto \mathbb{R}^2$  be a planar curve. Then the length of  $\gamma$  is given by*

$$l(\gamma) = 1/2 \iint_{\mathbb{P}} \eta_{\gamma}(\rho, \theta) \, d\rho d\theta$$

where  $\mathbb{P} = \mathbb{R}^+ \times [0, 2\pi[$  and for all  $(\rho, \theta) \in \mathbb{P}$ ,  $\eta_{\gamma}(\rho, \theta) = |\gamma([0, 1]) \cap D(\rho, \theta)| \in \mathbb{N} \cup \{\infty\}$  which is the number of intersection points of the curve  $\gamma$  with the straight line  $D(\rho, \theta)$  as represented in Figure 7.

Then, if  $\eta_E(\rho, \theta)$  is the number of points of  $E$  on the straight line  $D(\rho, \theta)$ , we have

$$\mathcal{L}(E) = 1/2 \iint_{\mathbb{P}} \eta_E(\rho, \theta) \, d\rho d\theta$$

Yet any straight line intersects a convex set in 0, 1 (if it is a tangent line) or 2 points.

With one point source we have :

**Proposition 4.** *Given a convex set  $E \subset \mathbb{R}^2$  and a point source  $S \in \mathbb{R}^2$ . The perimeter  $\mathcal{L}(E)$  of  $E$  necessarily verifies the following inequality:*

$$\mathcal{L}^2(E) \geq \iint_{I_{\rho}, \text{Supp}_{\mathbb{R}}(E, S)} d\rho d\theta = |I_{\rho}| * |\text{Supp}_{\mathbb{R}}(E, S)| \geq 1/2 |\cos(\theta_u) - \cos(\theta_d)| * |\theta_u + \theta_d|$$

Where  $I_{\rho} = [\rho_{min}, \rho_{max}]$ ,  $\theta_u$  and  $\theta_d$  as represented in Figure 8.

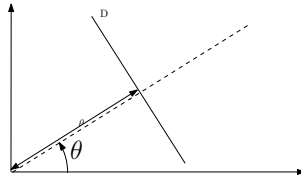


Fig. 7. Representation of a straight line with  $(\rho, \theta) \in \mathbb{R}^2 \times [0, 2\pi[$

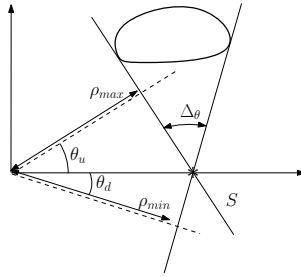


Fig. 8. Representation of the rays of a point source  $S$ .  $\Delta\theta = \theta_u + \theta_p$

### 4.2 Higher Bound of the Perimeter

To find a higher bound of the perimeter of a given convex set  $E \subset \mathbb{R}^2$ , we need two point sources  $S$  and  $S'$  of  $\mathbb{R}^2$ . Similarly to the notation for  $S$ , we denote  $\theta'$  the angles of the rays issuing from  $S'$ .

Let  $\theta_1, \theta_n, \theta'_1$  and  $\theta'_m$  be such that

$$\theta_1 = \min \{ \theta \in \text{Supp}_{\mathbb{R}}(E, S) \}, \theta_n = \max \{ \theta \in \text{Supp}_{\mathbb{R}}(E, S) \},$$

$$\theta'_1 = \min \{ \theta' \in \text{Supp}_{\mathbb{R}}(E, S') \}, \theta'_m = \max \{ \theta' \in \text{Supp}_{\mathbb{R}}(E, S') \}.$$

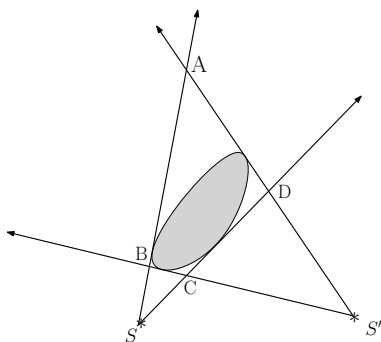
The following result is true only when both extreme rays of  $S$  ( $R_{\theta_1}$  and  $R_{\theta_n}$ ) intersect with both extreme rays of  $S'$  ( $R_{\theta'_1}$  and  $R_{\theta'_m}$ ). In this situation, let us consider  $A, B, C$  and  $D$  the intersection points of the extreme rays of  $S$  and  $S'$  see Figure 9. It is evident that we have  $E \subseteq ABCD$ . Thus we can prove the following result.

**Proposition 5.** *Let  $E \subset \mathbb{R}^2$  and two point sources  $S, S' \in \mathbb{R}^2$ . The perimeter of  $\mathcal{L}(E)$  of  $E$  necessarily verifies the following inequality:*

$$\mathcal{L}(E) \leq \mathcal{L}(ABCD)$$

where  $A, B, C$  and  $D$  are the intersection points of the extreme rays of  $S$  and  $S'$ .

*Proof.* As proven in [3], since  $ABCD$  is a convex,  $E$  is a convex as well, and  $E \subseteq ABCD$ , then  $\mathcal{L}(E) \leq \mathcal{L}(ABCD)$ .



**Fig. 9.**  $A, B, C$  and  $D$  the intersection points of the extreme rays of  $S$  and  $S'$

### 4.3 Example

We consider the same square  $E = [1, 2]^2$  as for the last section. We suppose that the vertices are  $(1, 1), (2, 1), (1, 2)$  and  $(2, 2)$ . We have  $\mathcal{L}(E) = 4$ .

– First lower bound:

Using the information about the surface area  $\mathcal{A}(E)$  we have:  $\mathcal{L}^2(E) \geq 4\pi \times 1 = 12.566$  and so  $\mathcal{L} \geq \sqrt{12.566} = 3.54$ .

– Second lower bound:

To apply the Crofton formula we consider a point source  $S = (2, 0)$  (see Figure 10).

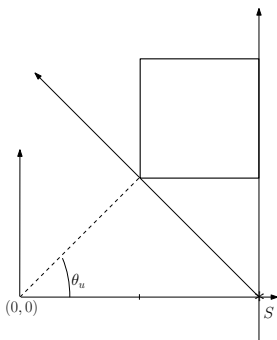
We have then  $\theta_u = 45$  and  $\theta_d = 0$ . Thus:

$$\mathcal{L}^2(E) \geq \frac{1}{2} \left| \frac{\sqrt{2}}{2} - 1 \right| * 45 = 6.59$$

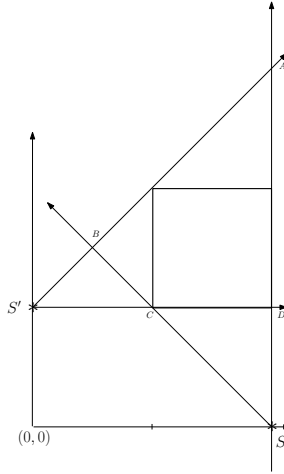
– Upper bound:

To compute the upper bound to the perimeter, we consider a second point source  $S' = (0, 1)$  as illustrated on Figure 11.

We have then  $\mathcal{L}(E) \leq 3 + 2\sqrt{2} = 5.82$ .



**Fig. 10.** The square  $E$  and a point source  $S = (2, 0)$ .  $\mathcal{L}(E) = 4$ .



**Fig. 11.** Computation of an upper bound to the square.  $ABCD = 3 + 2\sqrt{2}$

The perimeter verifies then :

$$3.54 \leq \mathcal{L}(E) = 4 \leq 5.82.$$

## 5 Conclusion

We presented in this paper a new method of extracting some information about sets given the projections from point sources. With one point source, we can estimate the surface area of the projected set and find two lower bounds for its perimeter. An additional point source is needed in order to have a higher bound of the perimeter. This quantitative information is deduced directly from the projection with no reconstruction step and with less point sources than for the reconstruction.

A question remains about the possibility of estimating the perimeter from projections with two point sources. Another interesting perspective to this work is the extraction of qualitative information from projections such as topological information.

## References

1. Abdmouleh, F., Daurat, A., Tajine, M.: Discrete Q-Convex Sets Reconstruction from Discrete Point X-Rays. In: Aggarwal, J.K., Barneva, R.P., Brimkov, V.E., Koroutchev, K.N., Korutcheva, E.R. (eds.) IWCIA 2011. LNCS, vol. 6636, pp. 321–334. Springer, Heidelberg (2011)
2. Balázs, P., Gara, M.: Decision Trees in Binary Tomography for Supporting the Reconstruction of  $h\nu$ -Convex Connected Images. In: Blanc-Talon, J., Bourennane, S., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2008. LNCS, vol. 5259, pp. 433–443. Springer, Heidelberg (2008)

3. Baudrier, É., Tajine, M., Daurat, A.: Convex-Set Perimeter Estimation from Its Two Projections. In: Aggarwal, J.K., Barneva, R.P., Brimkov, V.E., Koroutchev, K.N., Korutcheva, E.R. (eds.) IWCIA 2011. LNCS, vol. 6636, pp. 284–297. Springer, Heidelberg (2011)
4. Osserman, R.: The isoperimetric inequality. *B. Am. Soc.* 84, 1182–1238 (1978)
5. Van Dalen, B.: Boundary length of reconstructions in discrete tomography. *SIAM J. Discrete Math.* 25, 645–659 (2011)
6. Volčič, A.: A three-point solution to Hammer's X-ray problem. *J. London Math. Soc.* 34, 349–359 (1986)

# On the Non-additive Sets of Uniqueness in a Finite Grid

Sara Brunetti<sup>1</sup>, Paolo Dulio<sup>2</sup>, and Carla Peri<sup>3</sup>

<sup>1</sup> Dipartimento di Scienze Matematiche e Informatiche, Università di Siena,  
Pian dei Mantellini 44, 53100, Siena, Italy

sara.brunetti@unisi.it

<sup>2</sup> Dipartimento di Matematica “F. Brioschi”, Politecnico di Milano,  
Piazza Leonardo da Vinci 32, I-20133 Milano, Italy

paolo.dulio@polimi.it

<sup>3</sup> Università Cattolica S.C.,

Via Emilia Parmense, 84 29122 Piacenza, Italy

carla.peri@unicatt.it

**Abstract.** In Discrete Tomography there is a wide literature concerning (weakly) bad configurations. These occur in dealing with several questions concerning the important issues of uniqueness and additivity. Discrete lattice sets which are additive with respect to a given set  $S$  of lattice directions are uniquely determined by  $X$ -rays in the direction of  $S$ . These sets are characterized by the absence of weakly bad configurations for  $S$ . On the other side, if a set has a bad configuration with respect to  $S$ , then it is not uniquely determined by the  $X$ -rays in the directions of  $S$ , and consequently it is also non-additive. Between these two opposite situations there are also the non-additive sets of uniqueness, which deserve interest in Discrete Tomography, since their unique reconstruction cannot be derived via the additivity property. In this paper we wish to investigate possible interplays among such notions in a given lattice grid  $\mathcal{A}$ , under  $X$ -rays taken in directions belonging to a set  $S$  of four lattice directions.

2000 *Mathematics Subject Classification.* Primary 05D05; Secondary 05A17; 11P81.

**Keywords:** Additivity, bad-configuration, reconstruction, uniqueness, weakly bad configuration.

## 1 Introduction

In Discrete Tomography the usual line integrals employed in Computerized Tomography are replaced simply by the discrete  $X$ -rays, counting the number of points on each line parallel to given directions, so providing the so-called Discrete Radon Transform (DRT). The inversion of DRT aims to deduce the local atomic structure from the collected counting data. The original motivation came from High-Resolution Transmission Electron Microscopy (HRTEM) which is able to obtain images with atomic resolution and provides quantitative information on



the number of atoms that lie in single atomic columns in crystals choosing main  $X$ -ray directions such as  $(0, 1)$ ,  $(1, 0)$ ,  $(1, 1)$ ,  $(1, 2)$ ,  $\dots$  to be resolvable by the microscopy (see [18–20]). The high energies required to produce the discrete  $X$ -rays of a crystal mean that only a small number of  $X$ -rays can be taken before the crystal is damaged. Therefore, DT focuses on the reconstruction of images with few different grey levels, and, in particular, on the reconstruction of binary images from a small number of  $X$ -rays. It is worth mentioning that this problem was considered in its pure mathematical form even before its connection with electron microscopy ([6]). Atoms are modeled by lattice points and so crystals by finite sets of lattice points. The tomographic grid is a finite set  $\mathcal{G}$  of lattice points which are intersections of lines parallel to the  $X$ -ray directions corresponding to nonzero  $X$ -ray and feasible solutions of the reconstruction problem are subsets of  $\mathcal{G}$  [15]. If there is only one solution the lattice set is  $\mathcal{G}$ -unique, or simply unique. On this regard, a special class of geometric objects, called *additive* sets, has been studied in considerable depth (see Section 2 for the formal definition). It was shown in [6] that a finite subset  $F$  of  $\mathbb{Z}^2$  is uniquely determined by its  $X$ -rays in the coordinate directions if and only if  $F$  is additive. The sufficient condition was later extended to any dimension, pointing out that notions of additivity and uniqueness are equivalent when two directions are employed, whereas, for three or more directions, additivity is more demanding than uniqueness. Actually, every additive set is uniquely determined, but there are non-additive sets of uniqueness [7]. Further generalizations have been considered in [8], where the notion of additivity has been extended to  $n$ -dimension, with respect to a set of linear manifolds. The literature suggests that, without the additivity property, it may be quite difficult to decide whether a lattice set is uniquely determined by its  $X$ -rays taken in a set of more than three directions. In fact, the inversion of DRT is generally NP-hard ([17]), so that any reconstruction algorithm must consist of exponentially many steps in the size of  $F$ . One related problem is to find suitable sub-classes of lattice sets that can be reconstructed in polynomial time (see, for instance [2, 3]), or to provide uniqueness results from the a priori knowledge of the geometric features of the class (see [9, 10]). In general, uniqueness is not a property of the set  $S$  of  $X$ -ray directions, as for each  $S$  there exists a lattice set which is not uniquely determined by  $S$ . On the contrary if we restrict to bounded sets in a given rectangular grid  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$ , there are whole families of lattice directions which uniquely determine all the finite subsets (so called bounded sets) of  $\mathcal{A}$  [4]. Unless the tomographic grid  $\mathcal{G}$  is contained in  $\mathcal{A}$ , uniqueness in  $\mathcal{A}$  does not imply uniqueness in  $\mathcal{G}$ . Therefore it is interesting to try to understand which bounded sets uniquely determined by  $S$  in  $\mathcal{A}$  are also  $\mathcal{G}$ -unique and/or additive. In some sense, roughly speaking, we measure “how strongly unique”, they are. In Section 4 we classify all the bounded sets in a grid  $\mathcal{A}$  which are uniquely determined by a set of four directions of uniqueness for  $\mathcal{A}$  that contains the coordinate directions. In this important case, we also compute the proportion of non-additive sets of uniqueness with respect to additive sets in  $\mathcal{A}$ . This ratio depends only on the number of  $X$ -ray directions whereas it is independent on the size of  $\mathcal{A}$ , and hence it is constant in the case

study. Finally, we show how to explicitly construct non-additive sets of uniqueness. These results partially answer an open question posed by Fishburn and Shepp in [8].

## 2 Background

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*. By *lines with direction*  $(a, b) \in \mathbb{Z}^2$  we mean lattice lines defined in the  $x, y$  plane by equations of the form  $ay = bx + t$ , where  $t \in \mathbb{Z}$ . We refer to a finite subset of  $\mathbb{Z}^2$  as a *lattice set*.

Let  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$  be a finite grid. We refer to the lattice sets  $E \subseteq \mathcal{A}$  as *bounded sets*. A bounded set  $E \subset \mathcal{A}$  can be identified, in a natural way, with its characteristic function  $\chi_E : \mathcal{A} \rightarrow \{0, 1\}$  defined by  $\chi_E(i, j) = 1$  for  $(i, j) \in E$ , and  $\chi_E(i, j) = 0$  otherwise.

For a function  $f : \mathcal{A} \rightarrow \mathbb{Z}$ , we write  $|f| = \max_{(i,j) \in \mathcal{A}} \{|f(i, j)|\}$ . Further, the *line sum* of  $f$  along the lattice line with equation  $ay = bx + t$  is defined as  $\sum_{aj=bi+t} f(i, j)$ .

Given a lattice direction  $(a, b)$ , the *X-ray* of  $E$  in the direction  $(a, b)$  is the function giving the number of points in  $E$  on each line parallel to  $(a, b)$ . Two sets  $E, F \subseteq \mathbb{Z}^2$  are said to be *tomographically equivalent* with respect to a set  $S$  of lattice directions if  $E$  and  $F$  have the same X-rays in the directions in  $S$ . A finite set  $E \subseteq \mathbb{Z}^2$  is a *set of uniqueness* with respect to a set  $S$  of lattice directions, or simply *S-unique*, if  $E$  is uniquely determined by its X-rays taken in the directions belonging to  $S$ . In other words, if  $F$  is tomographically equivalent to  $E$  with respect to  $S$ , then  $F = E$ . Given a finite set  $S$  of lattice directions, we say that two functions  $f, g : \mathcal{A} \rightarrow \{0, 1\}$  are *tomographically equivalent* if they have equal line sums along the lines corresponding to the directions in  $S$ . Note that two non trivial functions  $f, g : \mathcal{A} \rightarrow \{0, 1\}$  which are tomographically equivalent can be interpreted as characteristic functions of two lattice sets which are tomographically equivalent.

Let  $(a, b)$  be a lattice direction. Set

$$f_{(a,b)}(x, y) = \begin{cases} x^a y^b - 1, & \text{if } a > 0, b > 0 \\ x^a - y^{-b}, & \text{if } a > 0, b < 0 \\ x - 1, & \text{if } a = 1, b = 0 \\ y - 1, & \text{if } a = 0, b = 1. \end{cases}$$

Given a finite set  $S$  of lattice directions, we denote by  $F_S(x, y)$  the polynomial associated to  $S$  defined by (see [13, p. 19]):

$$F_S(x, y) = \prod_{(a,b) \in S} f_{(a,b)}(x, y) = \sum_{(i,j) \in \mathcal{A}} f(i, j) x^i y^j.$$

For any function  $h : \mathcal{A} \rightarrow \mathbb{Z}$ , its *generating function* is the polynomial defined by

$$G_h(x, y) = \sum_{(i,j) \in \mathcal{A}} h(i, j) x^i y^j.$$

Conversely, we say that the function  $h$  is *generated* by a polynomial  $P(x, y)$  if  $P(x, y) = G_h(x, y)$ . Notice that the function  $f$  generated by the polynomial  $F_S(x, y)$  vanishes outside  $\mathcal{A}$  if and only if the set  $S = \{(a_k, b_k)\}_{k=1}^d$  of  $d$  lattice directions satisfies the conditions

$$\sum_{k=1}^d a_k < m, \quad \sum_{k=1}^d |b_k| < n. \tag{1}$$

We then say that a set  $S = \{(a_k, b_k)\}_{k=1}^d$  of  $d$  lattice directions is *valid* for a finite grid  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$ , if (1) holds. Moreover, the function  $f$  generated by  $F_S(x, y)$  has zero line sums along the lines  $ay = bx + t$  taken in the directions  $(a, b) \in S$ . For example, if  $b \neq 0$

$$F_S(x, x^{-\frac{a}{b}}) = 0 = \sum_{(i,j) \in \mathcal{A}} f(i, j)x^i x^{-\frac{ia}{b}} = \sum_{\substack{(i,j) \in \mathcal{A} \\ a_j = bi + t}} f(i, j)x^{-\frac{t}{b}} = \sum_t x^{-\frac{t}{b}} \sum_{\substack{(i,j) \in \mathcal{A} \\ a_j = bi + t}} f(i, j).$$

The other cases can be obtained analogously.

Note that a generating function can be interpreted as follows: a monomial with its sign  $h(i, j)x^i y^j \in \mathbb{Z}[x, y]$  is associated to the lattice point  $\mathbf{p} = (i, j)$ , together with the weight  $h(i, j)$  which we call *multiplicity*. If  $|h(i, j)| > 1$  we say that  $\mathbf{p}$  is a *multiple point*. In order to simplify the notation furthermore we will denote a polynomial by  $P(x, y)$ , its associated lattice set by  $P$ , specifying the set of lattice points with positive (resp. negative) multiplicity by  $P^+$  (resp.  $P^-$ ). In particular we denote by  $F_S$  the set of lattice points associated to  $F_S(x, y)$ , counted with their multiplicities, namely

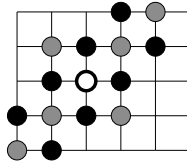
$$F_S = \{((i, j), l(i, j)) \in \mathbb{Z}^2 \times \mathbb{Z} : l(i, j) = f(i, j) \neq 0\}.$$

From the geometric point of view,  $F_S$  is a *S-weakly bad configuration*, namely a pair of lattice sets  $(Z, W)$  consisting of  $k$  lattice points  $z_1, \dots, z_k \in Z$ , and  $k$  points  $w_1, \dots, w_k \in W$ , not necessarily distinct (counted with multiplicity), such that for each direction  $(a, b) \in S$ , and for each  $z_r \in Z$ , the line through  $z_r$  in direction  $(a, b)$  contains a point  $w_r \in W$  (see Figure 1). We then say that a lattice set  $E$  has a weakly bad configuration if a weakly bad configuration  $(Z, W)$  exists for some  $k \geq 2$ , such that  $Z \subseteq E, W \subseteq \mathbb{Z}^2 \setminus E$ .

If all the points in the sets  $Z, W$  are distinct, then  $(Z, W)$  is called *S-bad configuration*. This notion plays a crucial role in investigating uniqueness problems, since any set  $S$  of directions is a set of uniqueness if and only if it has no bad configurations [6, Theorem 1].

Consider now the following definition. A finite set  $E \subseteq \mathbb{Z}^2$  is *additive with respect to S*, or simply *S-additive* if for each  $u \in S$ , there is a *ridge function*  $g_u$ , that is a function defined in  $\mathbb{Z}^2$  which is constant on each lattice line parallel to  $u$ , such that

$$E = \left\{ p \in \mathbb{Z}^2 : \sum_{u \in S} g_u(p) > 0 \right\}. \tag{2}$$



**Fig. 1.** The weakly bad configuration  $F_S$  associated to  $F_S(x, y) = (x - 1)(y - 1)(x^2y - 1)(xy^2 - 1) = x^4y^4 - x^3y^4 - x^4y^3 + x^3y^3 - x^2y^3 + xy^3 - x^3y^2 + 2x^2y^2 - xy^2 + x^3y - x^2y + xy - y - x + 1$  ( $S = \{(1, 0), (2, 1), (1, 2), (0, 1)\}$ ). The  $Z$  component is formed by the union of the white point (corresponding to  $f(2, 2) = 2$  and counted twice) with the set of grey points (which correspond to  $f(i, j) = 1$ ), while the set of black points (corresponding to  $f(i, j) = -1$ ) form the  $W$  component.

Additivity is also fundamental for treating uniqueness problems, due to the following facts (see [6, Theorem 2]):

1. Every additive set is a set of uniqueness.
2. There exist sets of uniqueness which are not additive.
3. A set is additive if and only if it has no weakly bad configurations.

### 3 Weakly Bad Configurations

L.Hajdu and R.Tijdman [14, Lemma 3.1] showed that if a function  $h : \mathcal{A} \rightarrow \mathbb{Z}$  has zero line sums along the lines taken in the directions in  $S$ , then  $F_S(x, y)$  divides  $G_h(x, y)$  over  $\mathbb{Z}$  ([14, Lemma 3.1]). In other words, since a weakly bad configuration is algebraically determined by  $h$ , we can reformulate the result as follows:

**Lemma 1.** *Let  $S$  be a set of lattice directions.  $G_h$  is an  $S$ -weakly bad configuration if and only if  $G_h(x, y) = H(x, y)F_S(x, y)$ , where  $H(x, y)$  is a polynomial.*

By Theorem 2.4 in [13], less than four directions are never sufficient to distinguish all the subsets of a given grid  $\mathcal{A}$ , and consequently  $|S| = 4$  represents a minimal choice for our problem. Therefore, throughout the paper, we assume  $S = \{u_1, u_2, u_3, u_4\}$ , with the further condition that  $u_1 + u_2 \pm u_3 = u_4$ . Motivation for this position relies on the fact that such cases give the unique situations where  $F_S$  represents a weakly bad configuration (see [4]). In particular,  $F_S$  has a multiple point, say  $p = \frac{1}{2}(u_1 + u_2 + u_3 + u_4)$ , and  $p$  is positive if  $u_4 = u_1 + u_2 - u_3$ , whereas it is negative if  $u_4 = u_1 + u_2 + u_3$ .

We next show that the weakly bad configuration  $F_S$  can be described by means of its multiple point and the given set  $S$  of directions. To this end we need some further notations.

We denote  $\hat{S} = \{\pm(v - u_4) : v \in \{u_1, u_2, u_4 - u_1 - u_2\}\}$ ,  $\pm S = \{\pm u : u \in S\}$  and  $D = \pm S \cup \hat{S}$ . Geometrically, the set  $D$  represents all the possible shifts which map the multiple point in  $F_S$  to another point of the weakly bad configuration  $F_S$  (see Figure 2), as it is shown by the following result.

**Proposition 1.** *The set  $F_S$  is completely determined by the couple  $(p, D)$ .*

*Proof.* Assume that  $b \geq 0$  for all  $(a, b) \in S$ , the positive points of  $F_S$  are

$$\begin{aligned} & \mathbf{0}, & u_1 + u_2, u_1 + u_3, u_1 + u_4, \\ & u_2 + u_3, u_2 + u_4, u_3 + u_4, u_1 + u_2 + u_3 + u_4 \end{aligned} \tag{3}$$

and the negative points are:

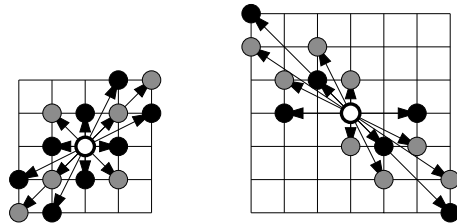
$$\begin{aligned} & u_1, & u_2, & u_3, & u_4, \\ & u_1 + u_2 + u_3, u_1 + u_2 + u_4, u_1 + u_3 + u_4, u_2 + u_3 + u_4, \end{aligned} \tag{4}$$

not all necessarily distinct. (We note that if  $b < 0$  for some  $(a, b) \in S$ , then the sets of positive and negative points exchange and are translated by the vector  $(0, -h)$ , where  $h$  is the sum of negative values of  $b$  for  $(a, b) \in S$ . In any case, the properties we are looking for are invariant by integer translations. From the algebraic viewpoint, this corresponds to substituting the polynomial  $F_S(x, y)$  by the rational function  $y^h F_S(x, y)$ .) By direct computation, we can check that all the points of the set  $F_S$  can be obtained as the translations of the multiple point  $p$  by a vector  $d$  in  $D$  and, if  $p = u_1 + u_2$  (i.e.  $f(p) > 0$ ), the multiplicities are given by:

$$\begin{aligned} l(p) &= f(p) = 2 \\ l(p + d) &= -(f(p) + f(d)) = -1, \text{ if } d \in \pm S \\ l(p + d) &= f(p) - f(d) = 1, \text{ if } d \in D \setminus \pm S. \end{aligned}$$

If  $p = u_1 + u_2 + u_3$  (i.e.  $f(p) < 0$ ), exchange the second and third cases and take the opposite signs. □

In general, uniqueness is not a property of the set  $S$  of X-ray directions, as for each  $S$  there exists a lattice set which is not uniquely determined by  $S$ . On the contrary if we restrict to bounded sets in a given rectangular grid  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$ , there are whole families of lattice directions which



**Fig. 2.** The set  $F_S$  is completely determined by the couple  $(p, D)$ , for  $p = (2, 2)$  and  $D = \pm\{(1, 0), (0, 1), (1, 2), (2, 1)\} \cup \pm\{(1, 1), (1, -1), (2, 2)\}$  (left) and for  $p = (3, 3)$  and  $D = \pm\{(0, 1), (1, -2), (2, -1), (3, -2)\} \cup \pm\{(1, -1), (2, 0), (3, -3)\}$  (right). The arrows show that each point of the configuration  $F_S$  can be reached from the multiple point by adding a vector in  $D$ .

uniquely determine bounded sets in  $\mathcal{A}$ . To show this, split  $D$  into two disjoint sets  $A, B$ , defined as follows.

$$A = \{(a, b) \in D : |a| > |b|\}, \tag{5}$$

and

$$B = \{(a, b) \in D : |b| > |a|\}. \tag{6}$$

Moreover, if  $|a| = |b|$ , for some  $(a, b) \in D$ , and  $\sum_{r=1}^4 a_r = h, \sum_{r=1}^4 |b_r| = k$  (where  $(a_r, b_r) = u_r$ ), we then include  $(a, b)$  in  $A$  if  $\min\{m - h, n - k\} = m - h$ , while  $(a, b) \in B$  otherwise. Thus we have  $D = A \cup B$ , where one of the sets  $A, B$  may be empty. Then we get the following Theorem 1, proved in [5].

**Theorem 1.** *Let  $S = \{u_1, u_2, u_3, u_4 = u_1 + u_2 \pm u_3\}$  be a valid set for the grid  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2, 0 \leq i < m, 0 \leq j < n\}$  and  $\sum_{r=1}^4 a_r = h, \sum_{r=1}^4 |b_r| = k$ , being  $(a_r, b_r) = u_r$  where  $r = 1, \dots, 4$ . Suppose that  $g : \mathcal{A} \rightarrow \mathbb{Z}$  has zero line sums along the lines in the directions in  $S$ , and  $|g| \leq 1$ . Then  $g$  is identically zero if and only if*

$$\min_{|a|} A \geq \min\{m - h, n - k\} \quad \text{and} \quad \min_{|b|} B \geq \min\{m - h, n - k\}, \tag{7}$$

and

$$m - h < n - k, \Rightarrow \forall (a, b) \in B \ (|a| \geq m - h \text{ or } |b| \geq n - k), \tag{8}$$

$$n - k < m - h, \Rightarrow \forall (a, b) \in A \ (|a| \geq m - h \text{ or } |b| \geq n - k), \tag{9}$$

where, if one of the sets  $A, B$  is empty, the corresponding condition in (7) drops.

Notice that Proposition 1, together with Lemma 1, suggests that in order to get a bad configuration one has to translate  $F_S$  so that its multiple point overlaps a point of  $F_S$  with multiplicity of opposite sign, without producing new multiple points. The above theorem provides the conditions for which such situations cannot occur within the grid  $\mathcal{A}$ .

## 4 Non-additive Sets of Uniqueness

By means of Theorem 1 we can check if a set of four directions  $S$  uniquely determines bounded sets in a grid  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$ . In general, any lattice set is unique (or  $\mathcal{G}$ -unique) if does not exists a different lattice set with the same  $X$ -rays in the tomographic grid  $\mathcal{G}$ . Unless  $\mathcal{G} \subseteq \mathcal{A}$ , uniqueness in  $\mathcal{A}$  does not imply uniqueness in the tomographic grid  $\mathcal{G}$ . Therefore it is interesting to try to understand which bounded sets, among those uniquely determined by  $S$  in  $\mathcal{A}$ , are also  $\mathcal{G}$ -unique and/or additive.

Trivially, the sets  $S$  non valid for  $\mathcal{A}$  uniquely determine all the subsets of  $\mathcal{A}$ , since no weakly bad configuration, and hence no bad configuration, can be constructed inside the grid. In general such bounded sets are not  $\mathcal{G}$ -unique, since  $\mathcal{G} \setminus \mathcal{A} \neq \emptyset$ . Differently, if  $\mathcal{G} \subseteq \mathcal{A}$  (for example if  $\{(1, 0), (0, 1)\} \in S$ , and  $S$  is non valid for  $\mathcal{A}$ ), then bounded sets are additive and therefore  $\mathcal{G}$ -unique.

Consider now a set  $S$  of four directions valid for  $\mathcal{A}$  which ensures uniqueness in the grid  $\mathcal{A}$ . As before, if  $\mathcal{G} \subseteq \mathcal{A}$ , then bounded sets are trivially  $\mathcal{G}$ -unique. Non-additive sets of uniqueness are those  $E$  among bounded sets such that  $F_S^+ \subseteq E$  (resp.  $F_S^- \subseteq E$ ) and  $E \cap F_S^- = \emptyset$  ( resp.  $E \cap F_S^+ = \emptyset$  ), because they have  $F_S$  as weakly bad configuration. More in general,

**Theorem 2.** *Let  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$ , and let  $S = \{u_1, u_2, u_3, u_4 = u_1 + u_2 \pm u_3\}$  be a set of uniqueness valid for  $\mathcal{A}$ . A set  $E \subset \mathcal{A}$  is non-additive if and only if there exists a polynomial  $P(x, y) = H(x, y)F_S(x, y)$ , such that  $|P| \geq 1$ ,  $P^+ \subseteq E$  and  $P^- \cap E = \emptyset$  (or  $P^- \subseteq E$  and  $P^+ \cap E = \emptyset$ ).*

*Proof.* Assume  $E \subset \mathcal{A}$  is non-additive with respect to  $S$ . Then  $E$  has a weakly  $S$ -bad configuration  $(Z, W)$ , where  $Z \subseteq E$  and  $W \cap E = \emptyset$  (or conversely). By [14, Lemma 3.1], we know that  $Z = P^+$  and  $W = P^-$  (or conversely), where  $P(x, y) = H(x, y)F_S(x, y)$  for some polynomial  $H(x, y)$ . Therefore  $P^+ \subset E$  and  $P^- \cap E = \emptyset$  (or  $P^- \subset E$  and  $P^+ \cap E = \emptyset$ ).

Assume now that a polynomial  $P(x, y) = H(x, y)F_S(x, y)$  exists, such that  $|P| \geq 1$ ,  $P^+ \subseteq E$  and  $P^- \cap E = \emptyset$  (or  $P^- \subseteq E$  and  $P^+ \cap E = \emptyset$ ). Then  $E$  has the weakly bad configuration  $(P^+, P^-)$ , so that it is non-additive. □

**Corollary 1.** *Assume  $\mathcal{A}$  and  $S$  are as in Theorem 2. Let  $P(x, y) = H(x, y)F_S(x, y)$  be any polynomial satisfying  $\deg_x P(x, y) < m$  and  $\deg_y P(x, y) < n$ . Then the set  $E = P^+$  (or  $E = P^-$ ) is non-additive and it is uniquely determined in  $\mathcal{A}$ .*

*Proof.* Since  $\deg_x P(x, y) < m$  and  $\deg_y P(x, y) < n$ ,  $E \subset \mathcal{A}$ , so that  $E$  is unique within  $\mathcal{A}$ . Moreover,  $E$  has the bad configuration  $(P^+, P^-)$ , which is consequently a weakly bad configuration, and  $E$  is also non-additive. □

Differently, every nonempty lattice set  $E \subseteq F_S$ , with  $E \neq F_S^-, E \neq F_S^+$ , is additive. This immediately follows by the observation that the tomographic grid related to  $E$  is contained in the tomographic grid related to  $F_S$ .

We can use Corollary 1 to explicitly construct non-additive (bounded) sets of uniqueness.

- Algorithm 3.**
1. Select a set  $S$  according to Theorem 1.
  2. Compute the weakly bad configuration associated to  $F_S(x, y)$ .
  3. Select any polynomial  $H(x, y)$  such that  $P(x, y) = H(x, y)F_S(x, y)$  satisfies  $\deg_x P(x, y) < m$  and  $\deg_y P(x, y) < n$ .
  4. Take each set  $E \subset \mathcal{A}$  such that  $P^+ \subset E$  and  $P^- \cap E = \emptyset$  (or  $P^- \subset E$  and  $P^+ \cap E = \emptyset$ ).

*Example 1.* Assume  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < 8, 0 \leq j < 7\}$ , and let  $S = \{(0, 1), (1, -2), (2, -1), (3, -2)\}$ . Then  $S$  is a valid set for  $\mathcal{A}$ , and we have

$$\begin{aligned} S_1 - S_2 &= \{\pm(-3, 3), \pm(-2, 0), \pm(-1, 1)\} \\ D &= \{\pm(0, 1), \pm(1, -2), \pm(2, -1), \pm(3, -2), \pm(-3, 3), \pm(-2, 0), \pm(-1, 1)\} \\ A &= \{\pm(2, -1), \pm(3, -2), \pm(-2, 0)\} \\ B &= \{\pm(0, 1), \pm(1, -2)\} \cup \{\pm(-3, 3)\} \\ \min\{m - h, n - k\} &= 1. \end{aligned}$$

Therefore,  $\min_{|a|} A = 2 > \min\{m - h, n - k\}$ ,  $\min_{|b|} B = 1 = \min\{m - h, n - k\}$ , and  $|a| \geq m - h = 2$  for all  $(a, b) \in A$ , so that conditions of Theorem 5 in [5] are satisfied. Consequently,  $S$  is a set of uniqueness for  $\mathcal{A}$ .

Assume  $H(x, y) = x - 1$ , and consider the following polynomial

$$\begin{aligned} P(x, y) &= (x - 1)F_S(x, y) = (x - 1)(y - 1)(x - y^2)(x^2 - y)(x^3 - y^2) = \\ &= x^7y - x^7 - x^6y^3 + x^6y^2 - x^6y + x^6 + x^5y^3 - \boxed{2x^5y^2} + x^5y + x^4y^4 - \\ &- \boxed{2x^4y^3} + \boxed{2x^4y^2} - x^4y + x^3y^5 - \boxed{2x^3y^4} + \boxed{2x^3y^3} - x^3y^2 - x^2y^5 + \\ &+ \boxed{2x^2y^4} - x^2y^3 - xy^6 + xy^5 - xy^4 + xy^3 + y^6 - y^5. \end{aligned}$$

Note that several multiple points appear. Let  $E$  be the set of lattice points corresponding to the monomials with positive sign. Then  $E$  is a non-additive set  $S$ -unique in  $\mathcal{A}$ . The same holds for the set of lattice points corresponding to the monomials with negative sign.

We now consider the special case where  $\{(1, 0), (0, 1)\} \subset S$  ( $S$  is a valid set of uniqueness for the given grid). Notice that all the non-additive sets of uniqueness are obtained as described above in Theorem 2 with  $H(x, y) = 1$ , whereas all the other bounded sets are additive. In fact, in the latter case, the bounded sets cannot have a weakly bad configuration since every weakly bad configuration is obtained by the product  $P(x, y) = H(x, y)F_S(x, y)$  for some  $H(x, y)$ , and hence possibly “enlarging”  $F_S$ . But in this case  $\deg_x P(x, y) \geq m$  or  $\deg_y P(x, y) \geq n$ , since  $\deg_x H(x, y) \geq 1$  or  $\deg_y H(x, y) \geq 1$  and  $m - h = n - k = 1$  ( $h, k$  as in Theorem 1).

Therefore, if  $\{(1, 0), (0, 1)\} \subset S$ , we get a complete classification of bounded sets. We can summarize the results as follows:

**Theorem 4.** *Let  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$ , and let  $S = \{u_1, u_2, u_3, u_4 = u_1 + u_2 \pm u_3\}$  be a set of uniqueness for  $\mathcal{A}$ , where  $\{(1, 0), (0, 1)\} \subset S$ .*

1. *If  $S$  is not valid for  $\mathcal{A}$ , then all bounded sets are additive.*
2. *If  $S$  is valid for  $\mathcal{A}$ , then:*
  - *every bounded set  $E$  such that  $F_S^+ \subseteq E \wedge E \cap F_S^- = \emptyset$ , or  $F_S^- \subseteq E \wedge E \cap F_S^+ = \emptyset$  is a non-additive set of uniqueness;*
  - *all the other bounded sets are additive.*



In [8] Fishburn et al. notice that an explicit construction of non-additive sets of uniqueness has proved rather difficult even though it might be true that non-additive uniqueness is the rule rather than exception. In particular they suggest that for some set of X-ray directions of cardinality larger than 2 the proportion of lattice sets  $E$  of uniqueness that are not also additive approaches 1 as  $E$  gets large. They leave it as an open question in the discussion section.

By means of Theorem 4 we can count the number of bounded additive and bounded non-additive sets of uniqueness as follows.

**Theorem 5.** *Let  $\mathcal{A} = \{(i, j) \in \mathbb{Z}^2 : 0 \leq i < m, 0 \leq j < n\}$ , and let  $S = \{u_1, u_2, u_3, u_4\}$  be a set of uniqueness, valid for  $\mathcal{A}$ , where  $\{(1, 0), (0, 1)\} \subset S$ . The proportion of bounded non-additive sets of uniqueness w.r.t. those additive is  $\frac{2}{2^{|F_S^-|} - 2}$ .*

*Proof.* The non-additive sets of uniqueness contained in  $F_S$  are precisely  $F_S^-$  and  $F_S^+$ , therefore  $\frac{2 \cdot 2^{|\mathcal{A} \setminus F_S|}}{2^{|\mathcal{A}|} - 2 \cdot 2^{|\mathcal{A} \setminus F_S|}} = \frac{2}{2^{|F_S^-|} - 2}$ . □

The theorem shows that for this family of sets of four directions, the proportion does not depend on the size of the lattice sets into consideration.

As a final remark, we note that a strong tool to treat these questions is provided by linear programming. Indeed the reconstruction problem can be reformulated as the following integer linear program (ILP):

$$Pz = q \text{ and } z \in \{0, 1\}^{m \times n}, \tag{10}$$

where the vector  $z$  represents the set  $\mathcal{A}$ , vector  $q$  contains the values of the line sums, and  $P$  is a 0, 1-matrix whose rows specify which points of  $\mathcal{A}$  are on each line  $l$  parallel to the directions in  $S$ . Each equation corresponds to a line sum on a lattice line  $l$ . The NP-hardness of the reconstruction problem for more than two directions is reflected in the integrality constraint for  $z$ . Therefore, in order to deal with this problem one can transform (10) by replacing the integrality constraint with an interval constraint, that is,  $z \in [0, 1]^{m \times n}$ . This approach has been introduced by Aharoni et al. in [1] and then studied by several authors (see, for instance [12, 21, 22]). To our aim, we can follow the method described in [7]. Indeed the authors proved that additive (lattice) sets are the unique solutions of the relaxed linear program (LP). Differently, for non-additive (lattice) sets there are many fuzzy sets with the given line sums, even if the lattice set solution is unique.

We refer the reader to the cited paper for a formal description of the method. However we recall that Fishburn and Shepp [8] choose to use interior point methods not only for efficiency “but because they produce solutions that lie in the center of optimality.” Moreover the set of all the solutions of LP is convex, that is, if  $z$  and  $z'$  are solutions, then  $\lambda z + (1 - \lambda)z'$  is a solution with  $0 < \lambda < 1$ . There follows that integral solutions are extreme members of the set. This allows to deduce that if the solution  $z$  of the relaxation determined by interior point method is such that for some  $i \in \{1, \dots, m \times n\}$   $z_i = 1$  (resp.  $z_i = 0$ ) then all the solutions  $z'$  have  $z'_i = 1$  (resp.  $z'_i = 0$ ).

Here we just sketch the possible outputs for our concern.

- 1) if LP does not admit any solution, then no solution exists for (10).
- 2) if the solution  $z$  of LP is such that  $z \in \{0, 1\}^{m \times n}$ , then it follows from the interior point method, convexity, and Theorems 1 and 2 of [6] that the corresponding lattice set is additive;
- 3) else if the solution  $z$  of LP is such that  $z \in [0, 1]^{m \times n}$ , then the situation is ambiguous and if there is exactly one extreme solution, then the corresponding lattice set is unique, even if non-additive.

Recently issues about uniqueness and additivity have been reviewed and settled by a more general treatment in [11]. In particular the concept of  $J$ -additivity has been introduced to study invariant sets, i.e. sets each point of which either belongs to all or does not belong to any solution of the reconstruction problem. As a further step in our research we would like to explore possible connections between our approach and the notion of  $J$ -additivity. We also plan to conduct some experiments to estimate, for a given grid  $\mathcal{A}$  and set  $S$  of uniqueness for  $\mathcal{A}$ , which bounded sets are additive and/or non-additive but unique, in the cases where the tomographic grid  $\mathcal{G}$  is not contained in  $\mathcal{A}$ .

## References

1. Aharoni, R., Herman, G.T., Kuba, A.: Binary vectors partially determined by linear equation systems. *Discr. Math.* 171, 1–16 (1997)
2. Brunetti, S., Daurat, A.: An algorithm reconstructing lattice convex sets. *Theoret. Comp. Sci.* 304, 35–57 (2003)
3. Brunetti, S., Daurat, A.: Reconstruction of convex lattice sets from tomographic projections in quartic time. *Theoret. Comput. Sci.* 406, 55–62 (2008)
4. Brunetti, S., Dulio, P., Peri, C.: Characterization of  $\{-1, 0, +1\}$  Valued Functions in Discrete Tomography under Sets of Four Directions. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) *DGCI 2011. LNCS*, vol. 6607, pp. 394–405. Springer, Heidelberg (2011)
5. Brunetti, S., Dulio, P., Peri, C.: Discrete Tomography determination of bounded lattice sets from four  $X$ -rays. *Discrete Applied Mathematics* (2012), doi:10.1016/j.dam.2012.09.010
6. Fishburn, P.C., Lagarias, J.C., Reeds, J.A., Shepp, L.A.: Sets uniquely determined by projections on axes II. Discrete case. *Discrete Math.* 91, 149–159 (1991)
7. Fishburn, P.C., Schwander, P., Shepp, L., Vanderbei, R.: The discrete Radon transform and its approximate inversion via linear programming. *Discrete Applied Math.* 75, 39–61 (1997)
8. Fishburn, P.C., Shepp, L.A.: Sets of uniqueness and additivity in integer lattices. In: Herman, G.T., Kuba, A. (eds.) *Discrete Tomography: Foundations, Algorithms and Application*, pp. 35–58. Birkhäuser, Boston (1999)
9. Gardner, R.J., Gritzmann, P.: Discrete tomography: Determination of finite sets by  $X$ -rays. *Trans. Amer. Math. Soc.* 349, 2271–2295 (1997)
10. Gardner, R.J., Gritzmann, P.: Uniqueness and complexity in discrete tomography. In: Herman, G.T., Kuba, A. (eds.) *Discrete Tomography: Foundations, Algorithms and Application*, pp. 85–113. Birkhäuser, Boston (1999)

11. Gritzmann, P., Langfeld, B., Wiegelmann, M.: Uniqueness in Discrete Tomography: three remarks and a corollary. *SIAM J. Discrete Math.* 25, 1589–1599 (2011)
12. Gritzmann, P., Prangenberg, D., de Vries, S., Wiegelmann, M.: Success and failure of certain reconstruction and uniqueness algorithms in discrete tomography. *Intern. J. of Imaging System and Techn.* 9, 101–109 (1998)
13. Hajdu, L.: Unique reconstruction of bounded sets in discrete tomography. *Electron. Notes Discrete Math.* 20, 15–25 (2005)
14. Hajdu, L., Tijdeman, R.: Algebraic aspects of discrete tomography. *J. Reine Angew. Math.* 534, 119–128 (2001)
15. Kuba, A., Herman, G.T.: Discrete tomography. In: *Appl. Numer. Harmon. Anal.* Birkhäuser, Boston (1999)
16. Kuba, A., Herman, G.T.: *Advances in Discrete Tomography and Its Applications.* In: *Appl. Numer. Harmon. Anal.* Birkhäuser, Boston (2007)
17. Irving, R.W., Jerrum, M.R.: Three-dimensional statistical data security problem. *SIAM J. Comput.* 23, 170–184 (1994)
18. Jinschek, J.R., Batenburg, K.J., Calderon, H., Van Dyck, D., Chen, F.R., Kisielowski, C.: Prospects for bright field and dark field electron tomography on a discrete grid. *Microscopy Microanal.* 10(suppl. 3), 44–45 (2004); *Cambridge Journals Online*
19. Kisielowski, C., Schwander, P., Baumann, F.H., Seibt, M., Kim, Y., Ourmazd, A.: An approach to quantitative high-resolution transmission electron microscopy of crystalline materials. *Ultramicroscopy* 58, 131–155 (1995)
20. Schwander, P., Kisielowski, C., Seibt, M., Baumann, F.H., Kim, Y., Ourmazd, A.: Mapping projected potential, interfacial roughness, and composition in general crystalline solids by quantitative transmission electron microscopy. *Phys. Rev. Lett.* 71, 4150–4153 (1993)
21. Weber, S., Schnoerr, C., Hornegger, J.: A linear programming relaxation for binary tomography with smoothness priors. *Electr. Notes Discr. Math.* 12 (2003)
22. Weber, S., Schüle, T., Hornegger, J., Schnörr, C.: Binary Tomography by Iterating Linear Programs from Noisy Projections. In: Klette, R., Žunić, J. (eds.) *IWCIA 2004*. LNCS, vol. 3322, pp. 38–51. Springer, Heidelberg (2004)

# On the Degree Sequences of Uniform Hypergraphs

Andrea Frosini, Christophe Picouleau, and Simone Rinaldi

Dipartimento di Sistemi e Informatica, Università di Firenze,  
Viale Morgagni, 65 - 50134 Firenze - Italy  
Laboratoire CEDRIC, CNAM  
292 rue St Martin, 75003 Paris - France

Dipartimento di Matematica e Informatica, Università di Siena,  
Pian dei Mantellini, 44 - 53100 Siena - Italy

andrea.frosini@unifi.it, christophe.picouleau@cnam.fr, rinaldi@unisi.it

**Abstract.** In hypergraph theory, determining a good characterization of  $d$ , the degree sequence of an  $h$ -uniform hypergraph  $\mathcal{H}$ , and deciding the complexity status of the reconstruction of  $\mathcal{H}$  from  $d$ , are two challenging open problems. They can be formulated in the context of discrete tomography: asks whether there is a matrix  $A$  with nonnegative projection vectors  $H = (h, h, \dots, h)$  and  $V = (d_1, d_2, \dots, d_n)$  with distinct rows.

In this paper we consider the subcase where the vectors  $H$  and  $V$  are both homogeneous vectors, and we solve the related consistency and reconstruction problems in polynomial time. To reach our goal, we use the concepts of Lyndon words and necklaces of fixed density, and we apply some already known algorithms for their efficient generation.

**Keywords:** Discrete Tomography, Reconstruction problem, Lyndon word, Necklace, hypergraph degree sequence.

## 1 Introduction

The degree sequence, also called graphic sequence, of a simple graph (a graph without loop or parallel edges) is the list of vertex degrees, usually written in nonincreasing order, as  $d = (d_1, d_2, \dots, d_n)$ ,  $d_1 \geq d_2 \geq \dots \geq d_n$ . The problem of characterizing the graphic sequences of graphs was solved by Erdős and Gallai (see [4]):

**Theorem 1. (Erdős, Gallai)** *A sequence  $d = (d_1, d_2, \dots, d_n)$  where  $d_1 \geq d_2 \geq \dots \geq d_n$  is graphic if and only if  $\sum_{i=1}^n d_i$  is even and*

$$\sum_{i=1}^k d_i \leq k(k-1) + \sum_{i=k+1}^n \min\{k, d_i\}, 1 \leq k \leq n.$$

An hypergraph  $\mathcal{H} = (V, \mathcal{E})$  is defined as follows (see [5]):  $Vert = \{v_1, \dots, v_n\}$  is a ground set of vertices and  $\mathcal{E} \subset 2^{Vert} \setminus \emptyset$  is the set of hyperedges such that  $e \not\subset e'$  for any pair  $e, e' \in \mathcal{E}$ . The degree of a vertex  $v \in Vert$  is the number of hyperedges  $e \in \mathcal{E}$  such that  $v \in e$ . An hypergraph  $\mathcal{H} = (Vert, \mathcal{E})$  is  $h$ -uniform

if  $|e| = h$  for all hyperedge  $e \in \mathcal{E}$ . Moreover  $\mathcal{H} = (Vert, \mathcal{E})$  has no parallel hyperedges, i.e.,  $e \neq e'$  for any pair  $e, e'$  of hyperedges. Thus a simple graph (loopless and without parallel edges) is a 2-uniform hypergraph.

The problem of the characterization of the degree sequences of  $h$ -uniform hypergraphs is one of the most relevant among the unsolved problems in the theory of hypergraphs [5] even for the case of 3-uniform hypergraphs. Also its complexity status is still open.

This problem has been related to a class of problems that are of great relevance in the field of discrete tomography. More precisely the aim of discrete tomography is the retrieval of geometrical information about a physical structure, regarded as a finite set of points in the integer lattice, from measurements, generically known as projections, of the number of atoms in the structure that lie on lines with fixed scopes. A common simplification is to represent a finite physical structure as a binary matrix, where an entry is 1 or 0 according to the presence or absence of an atom in the structure at the corresponding point of the lattice. One of the challenging problems in the field is then to reconstruct the structure, or, at least, to detect some of its geometrical properties from a small number of projections. One can refer to the books of G.T. Herman and A. Kuba [14,15] for further information on the theory, algorithms and applications of this classical problem in discrete tomography.

Here we recall the seminal result in the field of the discrete tomography due to Ryser [19]. Let  $H = (h_1, \dots, h_m), h_1 \geq h_2 \geq \dots \geq h_m$ , and  $V = (v_1, \dots, v_n), v_1 \geq v_2 \geq \dots \geq v_n$ , be two nonnegative integral vectors, and  $\mathcal{U}(H, V)$  be the class of binary matrices  $A = (a_{ij})$  satisfying

$$\sum_{j=1}^n a_{ij} = h_i \quad 1 \leq i \leq m \tag{1}$$

$$\sum_{i=1}^m a_{ij} = v_j \quad 1 \leq j \leq n \tag{2}$$

In this context  $H$  and  $V$  are called the row, respectively column, projection of  $A$ , as depicted in Fig. 1. Denoting by  $\bar{V} = (\bar{v}_1, \bar{v}_2, \dots)$  the conjugate sequence, also called the Ferrer sequence, of  $V$  where  $\bar{v}_i = |\{v_j : v_j \in V, v_j \geq i\}|$ . Ryser gave the following [19]:

**Theorem 2. (Ryser)**  $\mathcal{U}(H, V)$  is nonempty if and only if

$$\sum_{i=1}^m h_i = \sum_{i=1}^n v_i \tag{3}$$

$$\sum_{j=1}^i h_j \geq \sum_{j=1}^i \bar{v}_j \quad \forall i \in \{1, \dots, m\} \tag{4}$$

Moreover this characterization, and the reconstruction of  $A$  from its two projections  $H$  and  $V$ , can be done in polynomial time (see [14]). Some applications in discrete tomography requiring additional constraints can be found in [1,7,2,11,16,17,18,23].

As shown in [4] this problem is equivalent to the reconstruction of a bipartite graph  $G = (H, V, E)$  from its degree sequences  $H = (h_1, \dots, h_m)$  and  $V = (v_1, \dots, v_n)$ . Numerous papers give some generalizations of this problem for the graphs with colored edges (see [3,6,9,10,13]).

So, in this context, the problem of the characterization of the degree sequence  $(d_1, d_2, \dots, d_n)$  of an  $h$ -uniform hypergraph  $\mathcal{H}$  (without parallel edges) asks whether there is a binary matrix  $A \in \mathcal{U}(H, V)$  with nonnegative projection vectors  $H = (h, h, \dots, h)$  and  $V = (d_1, d_2, \dots, d_n)$  with distinct rows, i.e.,  $A$  is the incidence matrix of  $\mathcal{H}$  where rows and columns correspond to hyperedges and vertices, respectively. To our knowledge the problem of the reconstruction of a binary matrix with distinct rows has not been studied in discrete tomography.

In this paper, we carry on our analysis in the special case where the  $h$ -uniform hypergraph to reconstruct is also  $d$ -regular, i.e., each vertex  $v$  has the same degree  $d$ , in other words the vector of the vertical projection is homogeneous, i.e.  $V = (d, \dots, d)$ .

The studies in this paper focuses both on the decision problem, and on the related reconstruction problem, i.e. the problem of determining an element of  $\mathcal{U}(H, V)$  consistent with  $H$  and  $V$ . To accomplish this task we will design an algorithm that runs in polynomial time with respect to  $m$  and  $n$  the dimensions of the matrix to reconstruct. This algorithm uses the concepts of Lyndon words and necklaces of fixed density, also we apply some already known algorithms for their efficient generation.

## 2 Definitions and Introduction of the Problems

Let  $A$  be a binary matrix having  $m$  rows and  $n$  columns, and let us consider the two integer vectors  $H = (h_1, \dots, h_m)$  and  $V = (v_1, \dots, v_n)$  of its horizontal and vertical projections, respectively, as defined in Section 1 (see Fig. 1).

$$\begin{array}{r}
 V = \quad 3 \quad 1 \quad 2 \quad 2 \quad 6 \quad 3 \quad 2 \quad 4 \\
 \\
 \begin{array}{r}
 3 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1 \\
 2 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \\
 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \\
 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \\
 2 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \\
 H = \quad 7 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \\
 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \\
 2 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \\
 3 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \\
 3 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0
 \end{array}
 \end{array}$$

**Fig. 1.** A binary matrix, used in discrete tomography to represent finite discrete sets, and its vectors  $H$  and  $V$  of horizontal and vertical projections, respectively

These few notions are enough to state the general version of the problems we will consider in this paper (in fact, we consider more specialized versions, as defined further below):

**Consistency**  $(H, V, \mathcal{C})$

**Input:** two integer vectors  $H$  and  $V$ , and a class of discrete sets  $\mathcal{C}$ .

**Question:** does there exist an element of  $\mathcal{C}$  whose horizontal and vertical projections are  $H$  and  $V$ , respectively?

**Reconstruction**  $(H, V, \mathcal{C})$

**Input:** two integer vectors  $H$  and  $V$ , and a class of discrete sets.

**Task:** reconstruct a matrix  $A \in \mathcal{C}$  whose horizontal and vertical projections are  $H$  and  $V$ , respectively, if it exists, otherwise give failure.

In the sequel we are going to consider the class of binary matrices having no equal rows and homogeneous horizontal projections, denoted  $\mathcal{E}$ , due to its connections, as mentioned in the Introduction, with the characterization of the degree sequences of  $h$ -uniform hypergraphs.

In [20], Ryser gave a characterization of the instances of  $Consistency(H, V, \mathcal{C})$ , with  $\mathcal{C}$  being the class of the binary matrices, that admit a positive answer, after noticing that the following conditions are necessary for the existence of a matrix consistent with two generic vectors  $H$  and  $V$  of projections:

Condition 1: for each  $1 \leq i \leq m$  and  $1 \leq j \leq n$ , it holds  $h_i \leq n$  and  $v_j \leq m$ ;

Condition 2:  $\sum_{i=1}^m h_i = \sum_{j=1}^n v_j$ ,

and then he added a further condition in order to obtain the characterization of the instances that admit a solution, as recalled in the Introduction.

The authors of [8], pointed out that these two conditions turn out to be sufficient in case of homogeneous horizontal and vertical projections, by showing their maximality w.r.t. the cardinality of the related sets of solutions.

Ryser defined a well known greedy algorithm to solve  $Reconstruction(H, V, \mathcal{C})$  that does not compare the obtained rows, and does not admit an easy generalization to perform this further task.

So, since we want to restrict ourself to deal with matrices having different rows, and homogeneous horizontal projections, we approach the two problems in a different way, considering each row as a binary word, and grouping them into equivalence classes according to their cyclic shifts, as defined in the next section. Now, we state a third necessary condition for answering to  $Consistency(H, V, \mathcal{E})$ :

Condition 3:  $Consistency(H, V, \mathcal{E})$  has a negative answer if the dimension of the vector  $H$  is greater than  $\binom{n}{h}$ .

In other words, there does not exist a matrix having  $H$  and  $V$  as homogeneous projections, and more than  $\binom{n}{h}$  different rows; this result can be easily deduced using a cardinality argument. From our analysis, we will prove that the three conditions are also sufficient to solve in linear time the problem  $Consistency(H, V, \mathcal{E})$  with homogeneous  $H$  and  $V$ .

### 3 The Problem *Consistency*( $H, V, \mathcal{E}$ )

Let us face the consistency problem for the class  $\mathcal{E}$  on the homogeneous instance  $H$  and  $V$  from a different perspective, by observing that each row of a binary solution matrix can be regarded as a binary finite word  $u = u_1 u_2 \dots u_n$ , whose length  $n$  is the number of columns of the matrix, and whose number  $h$  of 1-elements is the common horizontal projection.

We note that applying a cyclic shift to the word  $u$ , denoted by  $s(u)$ , we obtain a different word  $s(u) = u_2 u_3 \dots u_n u_1$ , unless the cases  $u = (1)^n$  or  $u = (0)^n$ , of the same length, and having the same number of elements 1 inside, so it can be considered as a possible row different from  $u$  of a solution matrix. Iterating the shift of a word  $u$ , one can think to easily obtain a sequence of words that row wise arranged inside a matrix, lead to a solution of *Consistency*( $H, V, \mathcal{E}$ ). We indicate with  $s^k(u)$ , where  $k \geq 0$ , the application of  $k$  times the shift operator to the word  $u$ . Unfortunately, two problems may occur: the words repeat after at most  $n$  shifts, and the vertical projections may not reach the desired value  $v$ , i.e. the arranged shifts form a submatrix of a solution matrix (see Fig. 2). The following trivial property holds:

*Property 1.* Let  $u$  be a binary word of length  $n$  having  $h \leq n$  1-elements inside. Let us consider the  $n \times n$  matrix  $A$  obtained by row wise arranging all the cyclic shifts of  $u$ . Then,  $A$  has all horizontal and vertical projections equal to  $h$ .

One can easily notice that the rows of the matrix  $A$  may not all be different. Throughout the paper we will denote by  $M(u)$  the matrix obtained by row wise arranging all the different cyclic shifts of a word  $u$ . To establish how many different rows can be obtained by shifting a given binary word, we need to recall the definitions and main properties of necklaces and Lyndon words.

Following the notation in [21], a *binary necklace* (briefly *necklace*) is an equivalence class of binary words under cyclic shift. We identify a necklace with the lexicographically least representative  $u$  in its equivalence class, denoted by  $[u]$ .

The set of all (the words representative of) the necklaces with length  $n$  is denoted  $N(n)$ . For example,

$$N(4) = \{0000, 0001, 0011, 0101, 0111, 1111\}.$$

An important class of necklaces are those that are aperiodic. An aperiodic (i.e. period  $\geq n$ ) necklace is called a *Lyndon word*. Let  $L(n)$  denote the set of all Lyndon words with length  $n$ . For example,  $L(4) = \{0001, 0011, 0111\}$ .

We denote fixed-density necklaces, Lyndon words in a similar manner by adding the additional parameter  $d$  to represent the number of 1s in the words. We refer to the number  $d$  as the density of the word. Thus the set of necklaces with density  $d$  is represented by  $N(n, d)$ , and the set of Lyndon words with density  $d$  is represented by  $L(n, d)$ . For example,  $N(4, 2) = \{0011, 0101\}$ , and  $L(4, 2) = \{0011\}$ .



It is known from Gilbert and Riordan [12] that the number of fixed density necklaces and Lyndon words is

$$N(n, d) = \frac{1}{n} \sum_{j \mid \gcd(n, d)} \phi(j) \binom{n/j}{d/j}, \quad L(n, d) = \frac{1}{n} \sum_{j \mid \gcd(n, d)} \mu(j) \binom{n/j}{d/j}$$

respectively, where the symbols  $\phi$  and  $\mu$  refer to the Euler and Möbius functions. Now we enlighten the connection between these objects and our problem, refining Property 1.

**Proposition 1.** *If  $u$  is a word of length  $n$  and density  $h \leq n$ , then the cardinality of  $[u]$  (i.e. the number of rows of  $M(u)$ ) is a divisor of  $n$ .*

As a consequence, we have the following refinements:

**Proposition 2.** *If  $u$  is a Lyndon word of length  $n$  and density  $h$ , then the cardinality of  $[u]$  (i.e. the number of rows of  $M(u)$ ) is equal to  $n$ , and the vertical projections of  $M(u)$  are all equal to  $h$ .*

Figure 2 shows the 12 different cyclic shifts of the Lyndon word  $u = (0)^6(1)^6$  arranged in the first 12 rows of the matrix. All the rows of  $M(u)$  have horizontal and vertical projections equal to the density of  $u$ , i.e. 6.



**Fig. 2.** A solution to  $Reconstruction(H, V, C)$  when the horizontal projections have constant value 6, and the vertical projections 9. The submatrices  $M(u)$  obtained by row wise arranging the elements of three necklaces are highlighted. Note that  $M((0011)^3)$  and  $M((01)^6)$  are the only two possible necklaces of length 12 and density 6 having 4 and 2 rows, respectively.

**Proposition 3.** *If  $u = v^k$  (i.e.  $u = v \dots v$ ,  $k$  times), with  $k = \gcd\{n, h\}$ , is a necklace of length  $n$  and density  $h$ , and  $v$  a Lyndon word, then the cardinality of  $[u]$  is equal to  $n/k$ , and the vertical projections of  $M(u)$  are all equal to  $h/k$ .*

Figure 2 shows the  $12/3 = 4$  different cyclic shifts of the word  $u = (0011)^3$  arranged from row 13 to row 16 of the matrix, and the  $12/6 = 2$  different cyclic shifts of the word  $v = (01)^6$  in rows 17 and 18. All the rows of  $M(u)$  have horizontal projections equal to 6 and vertical projections equal to  $6/3 = 2$ , while the rows of  $M(v)$  have horizontal projections equal to 6 and vertical projections equal to  $6/6 = 1$ .

In the following we will prove that a pair  $H$  and  $V$  of projections satisfy Conditions 1, 2, and 3 if and only if they are consistent with a matrix in  $\mathcal{E}$ , solving *Consistency*( $H, V, \mathcal{E}$ ).

Let  $d_0 = 1, d_1, d_2, \dots, d_t$  be the increasing sequence of the common divisors of  $n$  and  $h$ . The following equation holds:

$$\binom{n}{h} = \sum_{i=0, \dots, t} \frac{n}{d_i} L\left(\frac{n}{d_i}, \frac{h}{d_i}\right).$$

This equation is an immediate consequence of the fact that each word of length  $n$  and density  $h$  belongs to exactly one necklace.

**Theorem 3.** *Let  $H$  and  $V$  be two homogeneous vectors of projections of dimension  $m$  and  $n$ , and elements  $h$  and  $v$ , respectively, satisfying Conditions 1, 2, and 3, i.e. being a valid instance of *Consistency*( $H, V, \mathcal{E}$ ). Then, there exists a Lyndon word  $L(n/d_i, h/d_i)$  such that  $n/d_i \leq m$ .*

*Proof.* Let us proceed by contradiction assuming that there does not exist a Lyndon word whose length is  $n/d < m$ , for each  $d \in d_1, \dots, d_t$ . Since  $H$  and  $V$  are homogeneous, and satisfy Conditions 1 and 2, then there exists a matrix  $A$  having  $H$  and  $V$  as projections (a consequence of Ryser’s characterization of solvable instances, as stated in [8], Theorem 3).

Let us assume that  $d = d_t = \gcd\{n, h\}$ ,  $h' = h/d$ , and  $n' = n/d$ ; by Condition 1, it holds

$$h'm = vn' \tag{5}$$

with  $n'$  and  $h'$  coprime, so  $v = h'(m/n')$ , and  $n'$  divides  $m$ . The hypothesis  $n' > m$  leads to a contradiction.  $\square$

Theorem 3 can be rephrased saying that if  $H$  and  $V$  are homogeneous consistent vectors of projections, then there exists a solution that contains all the elements of a necklace  $[u]$ . The solution in linear time of *Consistency*( $H, V, \mathcal{E}$ ) is a neat consequence:

**Corollary 1.** *Let  $H$  and  $V$  be two homogeneous vectors satisfying Conditions 1, 2, and 3. There always exists a matrix having different rows, and  $H$  and  $V$  as projections.*

The result of Theorem 3, together with the following proposition that point out a property of the necklace whose representant is  $u = (0)^{n-h}(1)^h$ , will be used in the next section to solve *Reconstruction*( $H, V, \mathcal{E}$ ).

**Proposition 4.** *Let  $u'$  be an element of the class  $[u]$ , with  $u = (0)^{n-h}(1)^h$ . The elements  $u', s^h(u'), s^{2h}(u'), \dots, s^{(k-1)h}(u')$ , with  $k = n/\gcd\{n, h\}$ , forms a subclass of  $[u]$ , and they can be arranged in a matrix  $A'$  such that*

1. *the vertical projections of  $A'$  are homogeneous and equal to  $h/\gcd\{n, h\}$ ;*
2.  *$A'$  is minimal with respect to the number of rows among the matrices having  $H$  as horizontal projections, and homogeneous vertical projections.*

The proof directly follows from the properties of the greatest common divisor. Let us denote with  $M(u)_1, \dots, M(u)_{\gcd\{n, h\}}$  the matrices defined in Proposition 4.

### 4 An Algorithm to Solve *Reconstruction*( $H, V, \mathcal{E}$ )

We start recalling that in [22] a constant amortized time (CAT) algorithm **Fast-FixedContent** for the exhaustive generation of necklaces of fixed length and density  $N(n, h)$  is presented. The author then shows that a slight modification of his algorithm can also be applied for the CAT generation of Lyndon words  $L(n, h)$  of fixed length and density. In particular, his algorithm –here denoted **GenLyndon**( $n, h$ )– constructs a generating tree of the words, and since the tree has height  $h$ , the computational cost of generating  $k$  words of  $L(n, h)$  is  $O(k \cdot h \cdot n)$ .

Our reconstruction algorithm works as follows:

**Rec**( $H, V, \mathcal{E}$ )

**Input :** Two homogeneous vectors:  $H = (h, \dots, h)$  of length  $m$ , and  $V = (v, \dots, v)$  of length  $n$ , satisfying Conditions 1, 2, and 3.

**Output :** An element of the class  $\mathcal{E}$  having  $H$  and  $V$  as horizontal and vertical projections, respectively.

**Step 1:** Let compute the sequence  $d_0 = 1 < d_1 < d_2 < \dots < d_t$  of the common divisors of  $n$  and  $h$ , and initialize the matrix  $A_{-1} = \emptyset$ .

**Step 2:** For  $i = 0$  to  $t$  do:

**Step 2.1:** By applying **GenLyndon**( $n, h$ ), generate the sequence of  $q = \min\{\lfloor v/h \rfloor, L(n, h)\}$  Lyndon words, denoted  $u_1, \dots, u_q$ . If  $q \neq L(n, h)$ , then do not include in the sequence the Lyndon word  $(0)^{n-h}(1)^h$ .

**Step 2.2:** Create the matrix  $A_i$ , obtained by row wise arranging the matrices  $M((u_j)^{d_i})$ , for  $j = 1, \dots, q$ .

Update  $v = v - q \cdot h$ .

If  $v = 0$  then output  $A_i$ ,

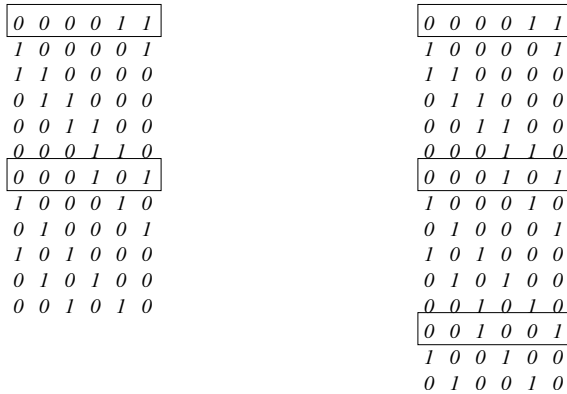
else if  $q \neq L(n, h)$ , create the matrix  $A$  obtained by row wise arranging the matrix  $A_i$  with the column wise arranging of  $d_i$  times the matrices  $M(u_j)$ , with  $u = (0)^{n-h}(1)^h$ ,  $j = 1, \dots, q'$ , and  $q' = v \cdot \gcd\{n, h\}/h$ ,

else update  $n = n/d_{i+1}$ , and  $h = h/d_{i+1}$ .

To better understand the reconstruction algorithm, we give a first simple example with the instance  $H = (2, \dots, 2)$  of length  $m = 15$ , and  $V = (5, \dots, 5)$  of length  $n = 6$ . In Step 1 the values  $d_0 = 1$ , and  $d_1 = 2$  are set.

In Step 2, **GenLyndon**(6, 2) generates  $q = 2$  Lyndon words, i.e. the words 000011, and 000101; since  $L(6, 2) = 2$ , then the word 000011 is included in the sequence. Now the matrix  $A_0$ , depicted in Fig. 3, on the left, is created. Finally, the values  $v = 5 - 2 \cdot 2 = 1$ ,  $n = 6/2 = 3$ , and  $h = 2/2 = 1$  are updated.

The second run of Step 2 starts, and **GenLyndon**(3, 1) generates the Lyndon word 001. The matrix  $A_1$  is created by row wise arranging  $A_0$  with the matrix  $M((001)^2)$  as shown in Fig. 3, on the right.



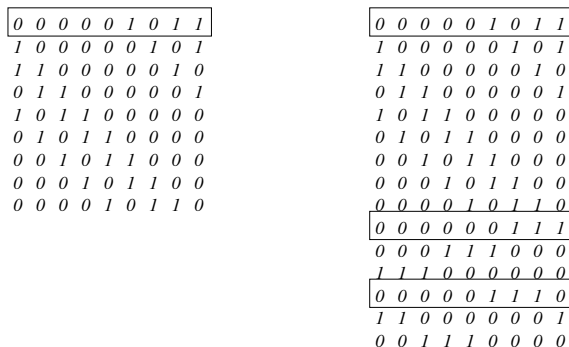
**Fig. 3.** The solution of dimension  $15 \times 6$  obtained by applying  $Rec(H, V, \mathcal{E})$  when the horizontal projections have constant value 2, and the vertical projections 5

A second example concerns the use of the word  $(0)^{n-h}(1)^h$  that in certain cases is set aside from the sequence of Lyndon words generated in Step 2: the instance we consider is  $H = (3, \dots, 3)$  of length  $m = 15$ , and  $V = (5, \dots, 5)$  of length  $n = 9$ . In Step 1 the values  $d_0 = 1$ , and  $d_1 = 3$  are set.

In Step 2, **GenLyndon**(9, 3) generates  $q = \min\{\lceil 5/3 \rceil, L(9, 3)\} = 1$  Lyndon words, i.e. the word 000001011; since  $q \neq L(9, 3)$ , then the word 000000111 is not included in the sequence. Now the matrix  $A_0$ , depicted in Fig. 4, on the left, is created. The value  $v = 5 - 3 \cdot 1 = 2$  is set.

Now, since  $q \neq L(9, 3)$ ,  $q' = 2(= 2 \cdot \gcd\{9, 3\}/3)$  submatrices of  $M(000000111)$  are computed, as defined in Proposition 4, and row wise arranged with  $A_0$ , obtaining the matrix in Fig. 4, on the right.

Note that without the use of the Lyndon word 000000111, the procedure is not able to reach the solution since in the second run of Step 2, **GenLyndon**(3, 1) generates only one Lyndon word, i.e. 001, whose matrix  $M((001)^3)$  has homogeneous vertical projections equal to 1, not enough to reach the desired value 2.



**Fig. 4.** The solution of dimension  $15 \times 9$  obtained by applying  $Rec(H, V, \mathcal{E})$  when the horizontal projections have constant value 3, and the vertical projections 5

The validity of  $Rec(H, V, \mathcal{E})$  is a simple consequence of Theorem 3. Clearly, the obtained matrix has homogeneous horizontal and vertical projections, equal to  $h$  and  $v$ , respectively, and, by construction, all the rows are distinct. Moreover, the algorithm always terminates since at each iteration, we add as many rows as possible to the final solution. Concerning the complexity analysis, we need to generate  $O(m)$  different Lyndon words and shift each of them  $O(n)$  times. So, since the algorithm **GenLyndon**( $n, h$ ) requires  $O(k \cdot h \cdot n)$  steps to generate  $k$  words of  $L(n, h)$ , the whole process takes polynomial time.

We conclude the paper by observing that our strategy could be suitably extended to a larger class of binary matrices with different rows, homogeneous horizontal projections, where the vertical projections are “quasi-homogeneous”, in the sense that their differences are bounded by a constant value.

## References

1. Balogh, E., Sorantin, E., Nyúl, L.G., Palágyi, K., Kuba, A., Werkgartner, G., Spuller, E.: Virtual dissection of the colon: technique and first experiments with artificial and cadaveric phantoms. In: Medical Imaging 2002: Image Processing, San Diego, USA. Proceedings of SPIE, vol. 4681, pp. 713–721 (2002)
2. Batenburg, K.J., Bals, S., Sijbers, J., Kuebel, C., Midgley, P.A., Hernandez, J.C., Kaiser, U., Encina, E.R., Coronado, E.A., Van Tendeloo, G.: 3D imaging of nano-materials by discrete tomography. Ultramicroscopy 109(6), 730–740 (2009)
3. Bentz, C., Costa, M.C., Picouleau, C., Ries, B., de Werra, D.: Degree-constrained edge partitioning in graphs arising from discrete tomography. Journal of Graph Algorithms and Applications 13(2), 99–118 (2009)
4. Berge, C.: Graphes. Gauthier-Villars, Paris (1983)
5. Berge, C.: Hypergraphs. North Holland (1989)
6. Brocchi, S., Frosini, A., Picouleau, C.: Reconstruction of binary matrices under fixed size neighborhood constraints. Theoretical Computer Science 406(1-2), 43–54 (2008)

7. Brocchi, S., Frosini, A., Rinaldi, S.: A reconstruction algorithm for a subclass of instances of the 2-color problem. *Theoretical Computer Science* 412, 4795–4804 (2011)
8. Brocchi, S., Frosini, A., Rinaldi, S.: The 1-Color Problem and the Brylawski Model. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 530–538. Springer, Heidelberg (2009)
9. Dürr, C., Guíñez, F., Matamala, M.: Reconstructing 3-colored grids from horizontal and vertical projections is NP-hard: A Solution to the 2-Atom Problem in Discrete Tomography. *SIAM J. Discrete Math.* 26(1), 330–352 (2012)
10. Gale, D.: A theorem on flows in networks. *Pacific J. Math.* 7, 1073–1082 (1957)
11. Gardner, R.J., Gritzmann, P.: Discrete tomography: determination of finite sets by X-rays. *Trans. Amer. Math. Soc.* 349, 2271–2295 (1997)
12. Gilbert, E.N., Riordan, J.: Symmetry types of periodic sequences. *Illinois J. Math.* 5, 657–665 (1961)
13. Guíñez, F., Matamala, M., Thomassé, S.: Realizing disjoint degree sequences of span at most two: A tractable discrete tomography problem. *Discrete Applied Mathematics* 159(1), 23–30 (2011)
14. Herman, G.T., Kuba, A.: *Discrete tomography: Foundations algorithms and applications*. Birkhauser, Boston (1999)
15. Herman, G.T., Kuba, A.: *Advances in Discrete Tomography and Its Applications*. Birkhauser, Boston (2007)
16. Irving, R.W., Jerrum, M.R.: Three-dimensional statistical data security problems. *SIAM Journal of Computing* 23, 170–184 (1994)
17. Matej, S., Vardi, A., Hermann, G.T., Vardi, E.: Binary tomography using Gibbs priors. In: Herman, G.T., Kuba, A. (eds.) *Discrete Tomography: Foundations, Algorithms and Applications*, pp. 191–212. Birkhauser, Boston (1999)
18. Prause, G.P.M., Onnasch, D.G.W.: Binary reconstruction of the heart chambers from biplane angiographic image sequence. *IEEE Transactions on Medical Imaging* 15, 532–559 (1996)
19. Ryser, H.: *Combinatorial Mathematics*. Mathematical Association of America and Quinn & Boden, Rahway (1963)
20. Ryser, H.J.: Combinatorial properties of matrices of zeros and ones. *Canadian Journal of Mathematics* 9, 371–377 (1957)
21. Ruskey, F., Sawada, J.: An efficient algorithm for generating necklaces with fixed density. *Siam J. Comput.* 29, 671–684 (1999)
22. Sawada, J.: A fast algorithm to generate necklaces with fixed content. *Theoret. Comput. Sci.* 301, 477–489 (2003)
23. Shliferstein, A.R., Chien, Y.T.: Switching components and the ambiguity problem in the reconstruction of pictures from their projections. *Pattern Recognition* 10, 327–340 (1978)

# How to Decompose a Binary Matrix into Three $hv$ -convex Polyominoes

Andrea Frosini and Christophe Picouleau

Dipartimento di Sistemi e Informatica, Università di Firenze,  
Viale Morgagni, 65 - 50134 Firenze - Italy  
Laboratoire CEDRIC, CNAM  
292 rue St Martin, 75003 Paris - France  
[andrea.frosini@unifi.it](mailto:andrea.frosini@unifi.it), [christophe.picouleau@cnam.fr](mailto:christophe.picouleau@cnam.fr)

**Abstract.** Given a binary matrix, deciding whether it can be decomposed into three  $hv$ -convex matrices is an  $\mathcal{NP}$ -complete problem, whereas its decomposition into two  $hv$ -convex matrices or two  $hv$ -polyominoes can be performed in polynomial time. In this paper we give a polynomial time algorithm that decomposes a binary matrix into three  $hv$ -polyominoes, if such a decomposition exists. These problems are motivated by the Intensity Modulated Radiation Therapy (IMRT).

**Keywords:** computational complexity, matrix decomposition, convex polyomino.

## 1 Introduction

The problem of the minimum decomposition of an integer matrix finds its practical applications in the Intensity Modulated Radiation Therapy (IMRT). For cancer treatment IMRT consists in delivering a radiation dose to destroy the tumor while maintaining the functionality of organs surrounding the tumor. The collimator is the medical machine that delivers the radiation dose (dose matrix). Technically, the collimator cannot deliver all shapes of matrices. The shape decomposition problem consists in decomposing the dose matrix into a set of deliverable shape matrices. In the literature, the decomposition into consecutive ones matrices is the most studied [1,5,6,12]. Baatar et al. [1] show that the problem of deciding whether a binary matrix  $A$  can be decomposed into at most  $K$   $h$ -convex matrices is  $\mathcal{NP}$ -complete in the strong sense even in the case where  $A$  has one single row.

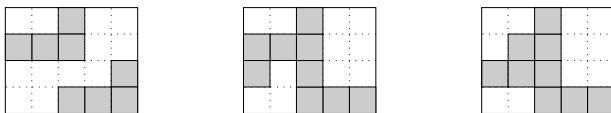
Motivated by a theoretical point of view, Jarray et al. [17] consider the problem of deciding whether it is possible to decompose a binary matrix into a (small) fixed number of binary matrices fulfilling some specific requirements as it is usual in the field of the theoretical discrete tomography. They show that the decomposition into three  $hv$ -convex matrices is  $\mathcal{NP}$ -complete in the strong sense whereas the decomposition problem into two  $hv$ -convex matrices is polynomial even in the case where each of the two matrices is an  $hv$ -polyomino. Note that this last requirement is of great importance for the applications in IMRT.

Following the work of Jarray et al. [17] we study the decomposition of a binary matrix into three  $hv$ -polyominoes. We design a polynomial time algorithm that given a binary matrix either returns a decomposition into three  $hv$ -polyominoes or *failure* if such a triplet of  $hv$ -polyominoes does not exist.

The paper is organized as follows: in the following section we give the definitions, and the basic notions about convex polyominoes. In Section 3 we give the polynomial time algorithm that decomposes a binary matrix into three  $hv$ -polyominoes if such a decomposition exists. In the last section we conclude and some directions for future works are given.

## 2 Definitions and Notations

Let us consider binary  $m \times n$  matrices and let us graphically represent them by sets of cells on a squared surface in a standard way, i.e. the 1-elements and 0-elements correspond to the presence or the absence of a cell in the related position. A binary matrix is *horizontally convex*, briefly  *$h$ -convex*, if the 1-elements of each row are connected; similarly, we say that it is *vertically convex*, briefly  *$v$ -convex*, if the 1-elements of each column are consecutive. Finally, an  $hv$ -matrix is a matrix that is horizontally and vertically convex (see Fig. 1).



**Fig. 1.** From the left to the right: a  $h$ -convex matrix, a  $v$ -convex polyomino, a  $hv$ -convex polyomino

A binary matrix is a *polyomino*, if its elements are connected w.r.t. the side adjacency relation. In the sequel, we consider matrices that are both  $hv$ -convex and polyominoes, i.e.  $hv$ -polyominoes. Polyominoes, first introduced in [14], are well known combinatorial objects, and some of their subclasses are widely studied both in Combinatorics, and in Discrete Tomography. In the first area, there are many enumeration results according to different parameters [4,7,8,10], that lead, among others, to efficient exhaustive and random generation algorithms [11,16], while in the second area, their geometrical aspects are studied by means of quantitative data, called projections, about the number of cells that lie on parallel lines having discrete directions [2,3] or that lie inside small areas of fixed shape [13] (for an overview of the topic see [15]).

We consider the rows and the columns of a  $m \times n$  matrix numbered in the standard way, i.e. downward from row 1 to row  $m$ , and rightward from column 1 to column  $n$ . These few notions are enough to state the problem *3-decomposition* we are going to study:

*given a binary matrix  $A$ , we want to efficiently decompose it into at most three  $hv$ -polyominoes, if possible, otherwise give failure.*



To accomplish this task we will define a polynomial time (w.r.t. the dimension of  $A$ ) algorithm  $3\text{-phv-DEC}$  that returns either at most three  $hv$ -polyominoes  $X$ ,  $Y$ , and  $Z$  such that  $A = X + Y + Z$  or *failure* when such a decomposition does not exist. Note that a strategy that fulfills this task gives also an answer in polynomial time to the related decision problem.

Our strategy will use the result of Proposition 1, that directly follows from an alternative definition of  $hv$ -polyomino: for any two elements  $(i, j)$  and  $(i', j')$  of a polyomino  $A$ , a *path* from  $(i, j)$  to  $(i', j')$  is a sequence  $(i_1, j_1), \dots, (i_r, j_r)$  of distinct adjacent elements of  $A$ , with  $(i, j) = (i_1, j_1)$  and  $(i', j') = (i_r, j_r)$ . For each  $1 \leq k \leq r - 1$ , we say that the two consecutive cells  $(i_k, j_k), (i_{k+1}, j_{k+1})$  form

- an *east* step if  $i_{k+1} = i_k$  and  $j_{k+1} = j_k + 1$ ;
- a *north* step if  $i_{k+1} = i_k - 1$  and  $j_{k+1} = j_k$ ;
- a *west* step if  $i_{k+1} = i_k$  and  $j_{k+1} = j_k - 1$ ;
- a *south* step if  $i_{k+1} = i_k + 1$  and  $j_{k+1} = j_k$ .

We define a path to be *monotone* if it is entirely made of only two of the four types of steps defined above. So, in [9], the following definition is given: a polyomino is  $hv$ -convex if and only if every pair of cells is connected by a monotone path.

**Proposition 1.** *Let  $(i, j)$  and  $(i', j')$  be two elements of the border of an  $hv$ -polyomino  $A$ . If we replace the path  $\pi$  that connects them and runs along the border of  $A$ , with any another monotone path  $\pi'$  that connects  $(i, j)$  to  $(i', j')$ , then the  $hv$ -convexity is preserved in the new polyomino.*

The intuitive idea of the replacement of the border path  $\pi$  with another monotone border path  $\pi'$  in  $A$  having the same starting and ending positions, is that of inserting a 1-element in each position of  $\pi'$ , then set to 0 the elements in the exterior of  $\pi'$ , w.r.t.  $A$ , and to 1 those inside  $A$ , the whole process has to be carried on without compromising the connectedness of  $A$ .

We define a monotone path connecting the cells  $a$  and  $b$  to be *maximum* [resp. *minimum*] if each of its elements has the maximum [resp. *minimum*] row index w.r.t. the element in the correspondent positions in all the other monotone paths that lead from  $a$  to  $b$  (note that, by definition, all the monotone paths that lead from  $a$  to  $b$  have the same number of elements).

### 3 A Fast Algorithm to Solve $3\text{-decomposition}$

The first step in defining a valid strategy to solve  $3\text{-decomposition}$  for a generic binary matrix  $A$ , consists in cutting off some trivial cases that allow us to immediately identify a solution or state that no solution exists.

So, we start by giving  $A$  as input to the algorithm  $2\text{-phv-DEC}$  defined in [17], and that solves  $2\text{-decomposition}$ . So, if  $2\text{-phv-DEC}$  decomposes  $A$ , then a solution to  $3\text{-decomposition}$  has been reached as well, otherwise, if it fails, we start the search for a decomposition into exactly three  $hv$ -polyominoes, if possible. It is straightforward that if the matrix  $A$  is composed by at least two non connected

components, then  $3$ -decomposition can be solved in polynomial time by means of a run of  $2$ -phv-DEC on each component.

As a consequence, the strategy to solve  $3$ -decomposition we are going to define, considers only input matrices  $A$  having one single connected component, i.e. polyominoes. It consists in the labeling of each cell of  $A$  with one label  $l \in \{x, y, z\}$ , according to the belonging to one of the three  $hv$ -polyominoes, indicated by  $X$ ,  $Y$  and  $Z$ , we are trying to split  $A$  into.

### 3.1 Starting the Labeling Process

We continue indicating with  $A$  the input polyomino of  $3$ -decomposition. First, we observe that the convexity of the polyominoes  $X$ ,  $Y$  and  $Z$  implies that if two elements of  $A$  with the same label lie on the same row [resp. column], then all the elements between them must belong to  $A$ , and share the same label. Second if two elements belong to two disconnected components of a row [resp. column], then they have different labels. During the labeling process we often use this property in order to assign labels *by convexity*.

We define a row of  $A$  to be a  $k$ -row if it is constituted by exactly  $k + 1$  non connected sequences of elements; the cells between two consecutive of these sequences form a *hole*; these same definitions can be given for columns (in Fig. 2 the two disconnected components of the extremal rows form two holes). By convexity, it is immediate to realize that if  $A$  has a  $k$ -row [resp.  $k$ -column] with  $k \geq 3$ , then  $3$ -decomposition has no solution.

**Proposition 2.** *If  $3$ -decomposition has a solution for  $A$ , then each 2-row of  $A$  has a labeling of the form*

$$0^{k_1}(L_1)^{k_2}0^{k_3}(L_2)^{k_4}0^{k_5}(L_3)^{k_6}0^{k_7},$$

*with  $k_2, \dots, k_6$  different from zero,  $k_1 + \dots + k_7 = n$ , and  $L_1$ ,  $L_2$ , and  $L_3$  being the three different labels.*

We say that in a labeling like that of Proposition 2 the labels alternate as  $L_1L_2L_3$ . It is noteworthy that the connectedness of the three polyominoes  $X$ ,  $Y$  and  $Z$  imposes that all the 2-rows [resp. 2-columns] of  $A$  have the same alternation of labels, see Fig. 2. This property can be extended by connectedness to two generic rows of  $A$  even if not the three labels are present, i.e. admitting one or more among the  $k_i$ s, with  $i = 2, 4, 6$ , to be zero:

**Proposition 3.** *Let  $i_1$  and  $i_2$  be two rows of  $A$ ,  $i_1$  being a 2-row. If  $3$ -decomposition has a solution for  $A$ , then  $i_1$  and  $i_2$  have the same alternation of the labels.*

This last proposition plays a special role in our decomposition strategy, since it allows to unambiguously assign an alternation of labels to the matrix  $A$ . Note that if no 2-rows are present in the matrix, then different alternations of variables can be possible, however, we underline that their number is six at most.

Finally, a similar definition of label alternation can be stated for the columns of  $A$ . So, from now on, we assume w.l.g. that  $A$  has the row alternation  $xyz$ .

Afterwards we fix one column alternation among the six permutations of  $\{x, y, z\}$  (we will require  $\beta$ - $phv$ -DEC to carry on the six combinations).

The labeling process we are going to define consists of two main steps, the first one takes care of the areas of  $A$  that lie between two 2-rows, and the second one that considers the remaining part of  $A$ , i.e. those areas that are delimited by at most one single 2-row.

**Labeling the Area between Two 2-Rows**

Consider the area  $2A$  between two 2-rows  $i_1$  and  $i_2$ , with  $i_1 < i_2$ , and let the leftmost and rightmost cells of the sequence of labels  $x$  in row  $i_1$  [resp.  $i_2$ ] be  $x_l^1$  and  $x_r^1$  [resp.  $x_l^2$  and  $x_r^2$ ]; identify in a similar way the extremal cells of the sequences of labels  $y$  and  $z$ , as shown in Fig. 2.

Since the border of  $A$  connects, on the left, the cells  $x_l^1$  and  $x_l^2$  of  $2A$ , and, on the right, the cells  $x_r^1$  and  $x_r^2$  of  $2A$ , it holds:

**Theorem 1.** *If 3-decomposition has a solution for  $A$ , then the elements of the border of  $A$  inside  $2A$  that connects the cells  $x_l^1$  and  $x_l^2$  [resp.  $x_r^1$  and  $x_r^2$ ] have label the  $x$  (see Fig. 2 (a)) [resp.  $z$ ].*

We assume that no 1-rows lie inside  $2A$ , and we label the cells of this area according to the mutual positions of the extremal labeled cells of the 2-rows:

**Proposition 4.** *If 3-decomposition has a solution for  $A$ , and the area  $2A$  has no 1-rows, then there also exists an (eventually different) decomposition such that the cells of  $2A$  till the border between  $X$  and  $Y$  are labeled as follows, according to the mutual positions of the extremal cells of the sequences  $x$  and  $y$  of the 2-rows: let us assume that the index of the column of  $y_l^1$  is greater than or equal to that of  $y_l^2$ , briefly  $y_l^1 \geq y_l^2$*

*Case i:  $x_r^1 \geq y_l^2$ . Each cell  $c \in 2A$  not already labeled such that*

1.  $c < y_l^2$ , has label  $x$ ;
2. it belongs to the minimum monotone path leading from  $y_l^2$  to  $y_l^1$ , has label  $y$ , as shown in Fig. 2, (a);

*Case ii:  $x_r^2 < x_r^1 < y_l^2$ . Each cell  $c \in 2A$  not already labeled such that*

1.  $c \leq x_r^1$ , has label  $x$ ;
2.  $x_r^1 < c < y_l^2$ , has the label  $x$  or  $y$  according to the column alternation of  $A$ ;
3. it belongs to the minimum monotone path leading from  $y_l^2$  to  $y_l^1$ , it has label  $y$ ;

*Case iii:  $x_r^1 < x_r^2$ . Each cell  $c \in 2A$  not already labeled such that*

1.  $c \leq x_r^2$ , has label  $x$ ;
2.  $x_r^2 < c < y_l^2$ , has the label  $x$  or  $y$  according to the column alternation of  $A$ ;
3. it belongs to the minimum monotone path leading from  $y_l^2$  to  $y_l^1$ , it has label  $y$ .

*Proof.* Case  $i$  is a direct consequence of Proposition 1, as shown in Fig. 2, (a).

Case  $ii$ : the labelings defined in 1., and 3. resemble those of Case  $i$ , while 2. is performed according to the column alternation of the labels, say  $L_1L_2L_3$ . The following cases arise:

- a. if at least one of the columns from the cell  $x_r^1$  to the cell  $y_l^2$  is a 2-column, then  $c$  has label  $L_2$ ;
- b. if the column immediately on the right of  $x_r^1$  has no cells above the area  $2A$ , then  $c$  has label  $x$ ;
- c. if the column immediately on the right of  $x_r^1$  has no cells below the area  $2A$ , then  $c$  has label  $y$ .

In all these three cases, we are not able to set a labeling that maintains, at the same time, the monotonicity of the right border of  $X$  and of the left border of  $Y$ , as in Case  $i$ . So, we determine the label of at least one cell inside the area defined in 2., and we extend it to all the remaining cells, obtaining the monotonicity of one of the two borders, and consequently preserving the convexity of both the polyominoes  $X$  and  $Y$ . In case  $a.$ , some cells inside the area of 2. already have the label  $L_2$  by the column alternation of the labels, in case  $b.$  (Fig. 2, (b)) the cells in the columns rightward  $x_r^2$ , and in row  $i_2 + 1$  can not have label  $x$ , so the cells inside the area 2. can have label  $x$  without compromising the convexity of the polyomino  $X$  in the solution, and leaving the path of the right border of  $Y$  monotone and minimum in  $2A$ . Finally, in case  $c.$  the cells in the columns leftward  $y_l^1$ , and in row  $i_1 - 1$  can not have label  $y$ , so the cells inside the area 2. can have label  $y$  without compromising the convexity of the polyomino  $Y$  in the solution, and leaving the path of the left border of  $X$  monotone in  $2A$ .

Case  $iii$  can be treated as Case  $ii$ . □

Note that the labeling of the right border of  $X$ , and consequently of the left border of  $Y$ , when the extremal cells  $x_r^1, y_l^1, x_r^2,$  and  $y_l^2$  of the 2-rows have the remaining three different mutual positions can be obtained by symmetry. Still by symmetry, we also obtain the labeling of the right border of  $Y$ , and consequently of the left border of  $Z$ , giving a complete labeling of the  $2A$  area when no 1-rows are present inside.

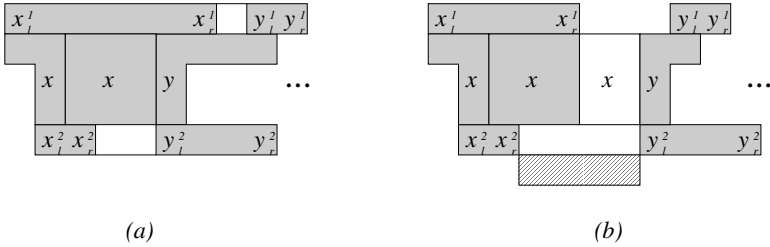
The labeling defined in Proposition 4 will be useful in the whole decomposition process, since it can be used each time two holes labeled as in Fig. 3, (a) (or, by extension, having the same labeling) have to be connected with a minimum monotone path.

Now we approach the problem of labeling the same area  $2A$  when 1-rows are present inside, by showing how to set the labels at the border of each of their holes, if not already provided, in order to connect them using Proposition 4; the row alternation is  $xyz$ , and the column alternation is fixed as well.

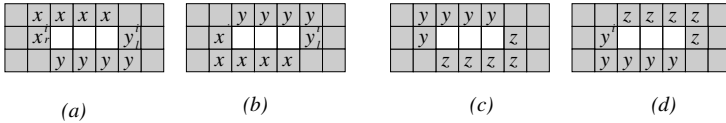
**Proposition 5.** *Let  $i$  be a 1-row of  $A$  lying in the  $2A$  area between the 2-rows  $i_1$  and  $i_2$ , that contains a hole from the column  $j$  to the column  $j'$ . If a solution to 3-decomposition for  $A$  exists, then there exists a solution such that the labeling of the elements of the border of the hole resembles one of those depicted in Fig.3, according to the alternation of the labels in the columns.*

The proof is a direct consequence of the convexity both of the rows and of the columns surrounding the hole.

We remark that each hole lies on the border between  $X$  and  $Y$  or  $Y$  and  $Z$ , but such a choice is not always deterministic, i.e. they can lie on different



**Fig. 2.** Two examples of labelings of the area  $2A$  for cases  $i$ , (a), and  $ii$ , (b). In (a) the labeling of the areas are set, from left to right, as follows: first by the labeling of the border of  $A$ , then by subcases 1, and 2. In (b) the labels are set, from left to right, as follows: the first area by the labeling of the border of  $A$ , then by subcases 1, 2, and 3. Note that, for subcase 2, is represented the situation  $b$ , where the cells above the hole in row  $i_1$  do not belong to  $A$ , and the dashed area below the hole of row  $i_2$  can not have any cell with label  $x$ .



**Fig. 3.** The labeling of the hole in (a) requires that the label  $x$  precedes the label  $y$  in the column alternation, while the labeling in (b) requires that the label  $y$  precedes the label  $x$ . Similarly it holds in (c) and (d) with the labels  $y$  and  $z$ . The four cases admit the row alternation  $xyz$ .

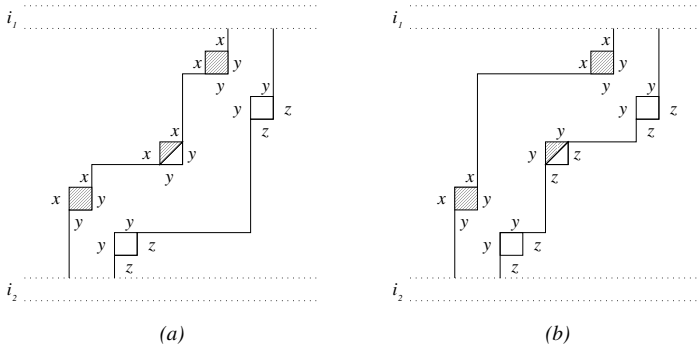
borders, for different solutions of *3-decomposition*. However, if some cells around it have already been labeled, then Fig. 3 shows how to extend these labels to the whole border of the hole. To carry on this labeling, we consider the mutual positions of all the holes inside  $2A$  and of the holes in the rows  $i_1$  and  $i_2$ .

Assuming that the column alternation of labels to be  $xyz$ , we arrange the holes of the 1-rows inside  $2A$  in a sequence  $\sigma = \{(i'_1, j'_1), \dots, (i'_k, j'_k)\}$ , according to the column indexes of their leftmost (void) cells, and if two cells lie in the same column, then they are arranged in decreasing order w.r.t. their row indexes. Note that the internal holes whose column indexes intersect those of the holes in the row  $i_1$  or  $i_2$  have already been labeled, since they are 2-columns, and the same holds for the holes that share at least one column. We recall that an element  $(i, j)$  of  $\sigma$  is a *left-to-right minimum* if for each element  $(i', j')$  that lies on its *left* in  $\sigma$ , it holds  $i \leq i'$ ; the definition of *right-to-left maximum* is similar.

**Proposition 6.** *If a solution of 3-decomposition for  $A$  exists, then the non already labeled hole  $(i'_t, j'_t)$ , with  $1 \leq t \leq k$ , has label as in Fig. 3, (a) if it is a left-to-right minimum, and not a right-to-left maximum (w.r.t. the row indexes) for the sequence  $\sigma$ .*

The proof follows from the convexity of the polyominoes  $X$ ,  $Y$ , and  $Z$ . We observe that, if in the column alternation of  $A$ , the label  $y$  precedes the label  $x$ , then the following changes has to be done: the sequence  $\sigma$  is defined on the rightmost cell of each hole, and in Proposition 6, the hole  $(i'_t, j'_t)$  is a right-to-left minimum and its labeling is as in Fig. 3, (b). A similar labeling can be defined for the holes that lie on the border between  $Y$  and  $Z$ . As a neat consequence of the proof of Proposition 6, it holds:

**Corollary 1.** *If there exists an element of  $\sigma$  that is neither a left-to-right minimum nor a right-to-left maximum, then 3-decomposition has no solution.*



**Fig. 4.** In (a), and (b) the two possible labelings of the holes internal to a  $2A$  area, when a hole that is both left-to-right minimum (dashed holes) and right-to-left maximum (blank holes) is present. The border between  $X$  and  $Y$  is the minimum one, while the one between  $Y$  and  $Z$  is the maximum.

Now we consider the case of a non labeled hole that is both left-to-right minimum and right-to-left maximum: we observe that it may allow two different labelings, i.e., with the assumed column alternation of labels, those of Fig. 3 (a) and (c), as shown in Fig. 4, by the half dashed hole.

However, if a hole is a left-to-right minimum and a right-to-left maximum, then there exists a hole on its right or on its left that is a left-to-right minimum or a right-to-left maximum, respectively. The label of the hole is the following:

**Proposition 7.** *Let the cell  $(i'_t, j'_t) \in \sigma$  be both a left-to-right minimum and a right-to-left maximum. If there exists a cell  $(i'_s, j'_s) \in \sigma$  that is a left-to-right minimum and not a right-to-left maximum, with  $j'_t < j'_s$ , then there exists a solution of 3-decomposition for  $A$  where the hole in  $(i'_t, j'_t)$  has its labeling as in Fig. 3 (a). If the cell  $(i'_s, j'_s)$  does not exist, then the hole in  $(i'_t, j'_t)$  has its labeling as in Fig. 3 (c).*

The proof is immediate since the presence of the left-to-right minimum  $(i'_s, j'_s)$  assures that the polyomino  $X$  extends till the column  $j'_t$ , since  $j'_t < j'_s$ .

A similar reasoning holds in the case that there exists a cell  $(i'_s, j'_s) \in \sigma$  that is a right-to-left maximum, and not a left-to-right minimum, with  $j'_t > j'_s$ .

This last observation completes the labeling process of the holes inside the  $2A$  area (when the label  $x$  precedes the label  $y$  in the column label alternation). So we can determine the border between  $X$  and  $Y$  by connecting, row after row from  $i_2$  to  $i_1$ , the holes labeled with  $x$  and  $y$  using monotone minimum pathes.

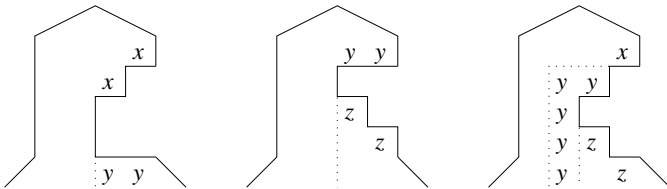
**Labeling the Zone with Only One 2-Row or without 2-Rows**

Properties 6 and 7 give a valuable way of detecting the borders of  $X$ ,  $Y$ , and  $Z$  in presence of holes, and below it will be used in a more general setting.

So, let us assume that the column alternation of labels in  $A$  is still  $xyz$  and that, w.l.g., row  $i$  is a 2-row, and, for each  $i' < i$ , row  $i'$  is not; we indicate this area with  $2A^-$  (in the sequel,  $2A^-$  can also indicate an area where row  $i$  is a 2-row, and, for each  $i' > i$ , row  $i'$  is not, or the whole matrix if there are no 2-rows). Now, we proceed in labeling  $2A^-$ : if no 1-rows are present, then we act similarly to what performed in Proposition 4, considering as holes also those inside the 1-columns along the border of  $A$ , say *border 1-columns*. Those holes that do not belong to any 1-row, by the connectedness of  $X$ ,  $Y$ , and  $Z$ , can lie only in this area, and their extremal cells must have different labels. Since we assumed the column alternation of labels to be  $xyz$ , then only three labelings are possible, i.e. those shown in Fig. 5. Consecutive border 1-columns whose holes share at least one row forms a *border 1-columns block*. The following is straightforward:

**Proposition 8.** *In a  $2A^-$  area, if the border 1-columns  $j$  and  $j'$  belong to two different blocks, then the extremal cells of their holes can not have the same labeling.*

By Property 8, it holds that the number of different labelings of the border 1-columns in a  $2A^-$  area is linear w.r.t. the number of columns, so we can consider each of them in a different line of computation, maintaining the polynomiality of the decomposition of  $A$ .



**Fig. 5.** The three possible labelings of blocks of border 1-columns in a  $2A^-$  area, when the column alternation is  $xyz$ . The possible borders of  $X$ ,  $Y$  and  $Z$  are set with dotted lines. Note that the rightmost example shows a situation where two different labelings of the holes are present in the same block.

If there also exist some 1-rows in  $2A^-$ , then we proceed in labeling them according to Properties 6 and 7. Now, all the holes inside  $2A^-$  are labeled, and the decomposition procedure can proceed by detecting the border between  $X$  and  $Y$ , and, using symmetry, the other one between  $Y$  and  $Z$ .

Finally, if the polyomino  $A$  contains neither 2-rows nor 2-columns, then the holes of its 1-rows, together with the holes of its border 1-columns, can be still labeled using the results of Properties 6 and 7.

### The Decomposition Procedure

Now, we are able to give the final procedure *3-phv-DEC* that detects a solution to *3-decomposition* on the polyomino  $A$ , if possible, by labeling its cells:

#### Procedure: *3-phv-DEC*

Input: a binary matrix  $A$ .

Output: three *hv*-polyominoes  $X$ ,  $Y$ , and  $Z$  such that  $A = X + Y + Z$ , if possible, otherwise *failure*.

Step 1: run the procedure *2-phv-DEC* defined in [17] on each connected component of  $A$ . If at most three *hv*-polyominoes are detected, then give them as output. Otherwise, if  $A$  has one single connected component, then goto Step 2, else give *failure*;

Step 2: label the 2-rows of  $A$  according to the row alternation  $xyz$ , and the 2-columns according to each of the six possible column alternation of labels. For each of them proceed to Step 3;

Step 3:

Step 3.1: for each  $2A$  area in  $A$ , label the border of  $A$  with  $x$  and  $z$ . If there are no 1-rows inside, then label the cells of  $X$  as defined in Proposition 4, and symmetrically do the same with  $Z$ , else label the holes as in Properties 6 and 7. If there is a non labeled hole, then give *failure*;

Step 3.2: for each  $2A^-$  area, label the holes of the 1-rows as in Properties 6 and 7, and for all possible combinations label the extremal cells of the holes of the border 1-columns according with the chosen column alternation. Note that this exhaustive labeling involves two different  $2A^-$  areas at most. Apply this labeling even in the case where  $A$  does not contain any 2-row.

Step 4: for each  $2A$  area inside the 2-rows  $i_1$  and  $i_2$ , consider the sequence  $\sigma$  as defined in Proposition 6. Connect with the minimum monotone path the holes of the 1-rows whose border contains  $x$  cells, acting as in Fig. 2, (a); the whole path will be part of the right border of  $X$ . Finally, connect the 2-row  $i_2$  with the monotone path as described in Proposition 4, after considering as  $x_r^1$  the cell  $(i'_s, j'_s)$  that is the first element of  $\sigma$  such that the cell  $(i'_s, j'_s - 1)$  has label  $x$ . The 2-row  $i_2$  can be connected to the monotone path similarly. Act symmetrically to detect the border between  $Y$  and  $Z$ ;

Step 5: for each  $2A^-$  area, act as in Step 4 connecting with the minimum monotone path the holes of the 1-rows whose border contains  $x$  cells, then connect the 2-row of  $2A^-$  with the monotone path. Complete the monotone path leading northward or southward, and maintaining its minimality (for



the northward case, see Fig. 2, (b) for an example). Symmetrically, detect the border between  $Y$  and  $Z$ . Finally, complete the labeling of the border of  $A$  in the area;

Step 6: if the polyomino  $A$  does not contain any 2-row, then connect the holes of the 1-rows as in Step 4 to find the border between  $X$  and  $Y$ , and, symmetrically, the border between  $Y$  and  $Z$ . Then complete the monotone path both northward and southward maintaining its minimality. Once found the internal border of the three polyominoes, complete the labeling of the cells at the border of  $A$ ;

Step 7: complete the labeling of the internal cells of  $A$  by convexity. If the three obtained polyominoes  $X$ ,  $Y$ , and  $Z$  are  $hv$ -convex, then give them as *output*, else give *failure*.

The correctness of the algorithm is assured by the fact that at each step we generate the border between  $X$  and  $Y$  by using minimum monotone pathes that connect all the detected  $x$  cells. Since those cells either have to belong to  $X$  for each solution of  $3$ -decomposition (Properties 4 and 6), or can be connected to them by a monotone path (so they belong to at least one solution of  $3$ -decomposition, as stated in Proposition 7), then the correctness follows. The same reasoning holds for the border between  $Y$  and  $Z$ . The computational complexity of the procedure is clearly polynomial, and it is performed for six times at most, one for each possible column alternation of the three labels. So the whole process turns out to be performed in polynomial time, as well.

## 4 Conclusions and Further Works

We designed a polynomial time algorithm that decomposes a binary matrix into three  $hv$ -polyominoes. Recall that in contrast the decomposition into three  $hv$ -matrices is an  $\mathcal{NP}$ -hard problem (see [17]). Now, a natural question is the following: given  $k > 3$ , can this algorithm be adapted for the decomposition into  $k$   $hv$ -polyominoes?

Other further works should be the following: motivated from a practical problem, give an extension of the algorithm in the three-dimensional case. Which kind of generalization should be the theoretical treatments in the higher dimensional discrete space.

## References

1. Baatar, D., Hamacher, H.W., Ehrgott, M., Woeginger, G.J.: Decomposition of integer matrices and multileaf collimator sequencing. *Discrete Applied Mathematics* 152(1-3), 6–34 (2005)
2. Barucci, E., Del Lungo, A., Nivat, M., Pinzani, R.: Reconstructing convex polyominoes from horizontal and vertical projections. *Theoretical Computer Science* 155, 321–347 (1996)
3. Barucci, E., Frosini, A., Rinaldi, S.: An algorithm for the reconstruction of discrete sets from two projections in presence of absorption. *Discrete Applied Mathematics* 151(1-3), 21–35 (2005)

4. Battaglino, D., Fedou, J.M., Frosini, A., Rinaldi, S.: Encoding Centered Polyominoes by Means of a Regular Language. In: Mauri, G., Leporati, A. (eds.) DLT 2011. LNCS, vol. 6795, pp. 464–465. Springer, Heidelberg (2011)
5. Boland, N., Hamacher, H., Lenzen, F.: Minimizing beam-on time in cancer radiation treatment using multileaf collimators. *Networks* 43(4), 226–240 (2003)
6. Bortfeld, T., Boyer, A., Kahler, D., Waldron, T.: X-ray field compensation with multileaf collimators. *International Journal of Radiation Oncology, Biology, Physics* 28(3), 723–730 (1994)
7. Bousquet-Mélou, M.: A method for the enumeration of various classes of column-convex polygons. *Discrete Mathematics* 154, 1–25 (1996)
8. Castiglione, G., Frosini, A., Munarini, E., Restivo, A., Rinaldi, S.: Enumeration of L-convex polyominoes. II. Bijection and area. In: Proceedings of FPSAC 2005, #49, pp. 531–541 (2005)
9. Castiglione, G., Restivo, A.: Reconstruction of L-convex Polyominoes. *Electronic Notes in Discrete Mathematics* 12, 290–301 (2003)
10. Delest, M., Viennot, X.: Algebraic languages and polyominoes enumeration. *Theoretical Computer Science* 34, 169–206 (1984)
11. Del Lungo, A., Duchi, E., Frosini, A., Rinaldi, S.: On the generation and enumeration of some classes of convex polyominoes. *The Electronic Journal of Combinatorics* 11, #R60 (2004)
12. Ehr Gott, M., Hamacher, H.W., Nußbaum, M.: Decomposition of Matrices and Static Multileaf Collimators: A Survey. In: Optimization in Medicine. Optimization and Its Applications, vol. 12, pp. 25–46 (2008)
13. Frosini, A., Nivat, M.: Binary Matrices under the Microscope: A Tomographical Problem. *Theoretical Computer Science* 370, 201–217 (2007)
14. Golomb, S.W.: Checker boards and polyominoes. *American Mathematical Monthly* 61(10), 675–682 (1954)
15. Herman, G.T., Kuba, A. (eds.): Discrete tomography: Foundations algorithms and applications. Birkhauser, Boston (1999)
16. Hochstätter, W., Loebel, M., Moll, C.: Generating convex polyominoes at random. In: Proceeding of the 5th FPSAC, *Discrete Mathematics*, vol. 153, pp. 165–176 (1996)
17. Jarray, F., Picouleau, C.: Minimum decomposition in convex binary matrices. *Discrete Applied Mathematics* 160, 1164–1175 (2012)
18. Shepard, D., Ferris, M., Olivera, G., Mackie, T.: Optimizing the delivery of radiation therapy to cancer patients. *SIAM Review* 41(4), 721–744 (1999)

# Multi-resolution Cell Complexes Based on Homology-Preserving Euler Operators

Lidija Čomić<sup>1</sup>, Leila De Floriani<sup>2</sup>, and Federico Iuricich<sup>2</sup>

<sup>1</sup> University of Novi Sad, Faculty of Technical Sciences  
comic@uns.ac.rs

<sup>2</sup> University of Genova, Department of Computer Science  
deflo@disi.unige.it,  
federico.iuricich@unige.it

**Abstract.** We have proposed a complete set of basis Euler operators for updating cell complexes in arbitrary dimensions, which can be classified as homology-preserving and homology-modifying. Here, we define the effect of homology-preserving operators on the incidence graph representation of cell complexes. Based on these operators, we build a multi-resolution model for cell complexes represented in the form of the incidence graph, and we compare its 2D instance with the pyramids of 2-maps, designed for images.

**Keywords:** geometric modeling, cell complexes, homology-preserving operators, multi-resolution representations.

## 1 Introduction

Cell complexes, together with simplicial complexes, have been used as a modeling tool in a variety of application domains. Several data structures have been designed in the literature for representing the connectivity of a cell complex (incidence and adjacency relations among the cells in the complex), such as incidence graphs, introduced in [6], and  $n$ -maps, introduced informally in [7].

Many topological operators have been designed for building and updating data structures representing 2D and 3D cell complexes. In [3], we have proposed a set of Euler operators which form a minimally complete basis for building and updating cell complexes in arbitrary dimensions in a topologically consistent manner. We distinguish between operators that preserve the homology of the complex, and the ones that modify it in a controlled manner. Homology-preserving operators add (or remove) a pair of cells of consecutive dimension, but they do not change the Betti numbers of the complex. Homology-modifying operators add (or remove) an  $i$ -cell, and increase (decrease) the  $i$ th Betti number.

Here, we define the effect of homology-preserving operators on the incidence graph, based on which we build a multi-resolution model for the topology of the complex, that we call the *Multi-Resolution Cell Complex (MCC)*. We present some experimental results validating the *MCC*, and we compare its 2D instance with the pyramidal model used for images represented in the form of a 2-map.

## 2 Background Notions

We review some notions on the topology of cell complexes (see [1] for details).

A  $k$ -cell in the Euclidean space  $\mathbb{E}^n$  is a homeomorphic image of an open  $k$ -dimensional ball, and a *cell  $d$ -complex* in  $\mathbb{E}^n$  is a finite set  $\Gamma$  of cells in  $\mathbb{E}^n$  of dimension at most  $d$ ,  $0 \leq d \leq n$ , such that (i) the cells in  $\Gamma$  are pairwise disjoint and (ii) for each cell  $\gamma \in \Gamma$ , the boundary of  $\gamma$  is a disjoint union of cells of  $\Gamma$ .

Intuitively, an  $n$ -dimensional quasi-manifold is an  $n$ -dimensional complex which can be obtained by gluing together  $n$ -cells along  $(n - 1)$ -cells (for details see [11]). In a quasi-manifold, an  $(n - 1)$ -cell belongs to the boundary of at most two  $n$ -cells. The notion of quasi-manifold is weaker than the notion of pseudo-manifold. Recall that a simplicial complex  $\Sigma$  is a *pseudo-manifold* if (i)  $\Sigma$  is homogenous (each simplex is a face of some  $n$ -simplex), (ii) each  $(n - 1)$ -simplex in  $\Sigma$  is an  $(n - 1)$ -face of at most two  $n$ -simplexes and (iii)  $\Sigma$  is strongly connected (for any two distinct  $n$ -simplexes  $\sigma$  and  $\tau$  in  $\Sigma$  there is a sequence  $\sigma = \sigma_1, \sigma_2, \dots, \sigma_k = \tau$ , such that  $\sigma_i$  and  $\sigma_{i+1}$  share an  $(n - 1)$ -simplex,  $1 \leq i < n$ ).

A variety of data structures have been proposed for representing the topology of cell complexes. Some represent the cells in the complex explicitly, e.g. incidence graphs, which can be used to represent arbitrary cell complexes, and abstract cellular complexes [9]. Some represent them implicitly, e.g.  $n$ -maps, which are used to represent orientable quasi-manifolds without boundaries.

An *Incidence Graph (IG)* [6] representing a cell complex  $\Gamma$  is a multigraph  $G = (N, A)$ , such that:

1. the set of nodes  $N$  is partitioned into  $n + 1$  subsets  $N_0, N_1, \dots, N_n$ , such that there is a one-to-one correspondence between the nodes in  $N_i$  (which we call  *$i$ -nodes*) and the  $i$ -cells of  $\Gamma$ ,
2. there are  $k$  arcs joining an  $i$ -node  $p$  with an  $(i + 1)$ -node  $q$  if and only if  $i$ -cell  $p$  appears  $k$  times on the boundary of  $(i + 1)$ -cell  $q$  in  $\Gamma$ .

We model the incidence (multi-)graph as an ordinary labeled graph, in which each node is labeled with the dimension of the corresponding cell, and each arc between two nodes is labeled with its multiplicity  $\varphi$  (the number of arcs between the two nodes in the corresponding multi-graph). If  $\Gamma$  is a simplicial complex then all the arcs in  $A$  are simple (with label equal to one).

An  *$n$ -map* (or  $n$ -dimensional *combinatorial map*) [2] is a finite set  $D$  of elements, called *darts*, together with  $n$  permutations  $\beta_i$  on  $D$ ,  $1 \leq i \leq n$ , such that  $\beta_i$  is an involution,  $2 \leq i \leq n$ , and  $\beta_i \circ \beta_j$  is an involution,  $i + 2 \leq j$ ,  $i, j \in \{1, \dots, n\}$ . Intuitively, a dart in  $D$  corresponds to an  $(n + 1)$ -tuple of cells  $(c_0, \dots, c_n)$ , where  $c_i$  is an  $i$ -cell,  $0 \leq i \leq n$ , and each  $c_i$  is on the boundary of  $c_{i+1}$ . For an  $n$ -map  $M = (D, \beta_1, \dots, \beta_n)$ ,  $n \geq 2$ , and a dart  $b$  in  $D$ , the 0-cell incident in  $b$  is the set of all darts that can be reached starting from  $b$  by applying any combination of permutations in the set  $\{\beta_1^{-1} \circ \beta_2, \dots, \beta_1^{-1} \circ \beta_n\}$ ; the  $i$ -cell incident in  $b$ ,  $1 \leq i \leq n$ , is obtained by applying permutations in  $\{\beta_1, \dots, \beta_n\} \setminus \{\beta_i\}$ . 2-maps are widely used for image processing and geometric modeling. In the 2D case, permutations  $\beta_1$  and  $\beta_2$  are usually denoted as  $\sigma$  and  $\alpha$ , respectively.

The Euler-Poincaré formula expresses the necessary validity condition of a cell complex with manifold or non-manifold carrier [1]. The Euler-Poincaré formula for a cell  $d$ -complex  $\Gamma$  with  $n_i$   $i$ -cells states that

$$\sum_{i=0}^d (-1)^i n_i = n_0 - n_1 + \dots + (-1)^d n_d = \sum_{i=0}^d (-1)^i b_i = b_0 - b_1 + \dots + (-1)^d b_d.$$

Here,  $b_i$  is the  $i$ th Betti number of  $\Gamma$ , and it measures the number of independent non-bounding  $i$ -cycles in  $\Gamma$ , i.e., the number of independent  $i$ -holes.

### 3 Related Work

A general idea of multi-resolution modeling is to provide several decompositions of a shape at different, uniform or variable, scales. We review related work on a hierarchical model for cell complexes, called *combinatorial* (or *n-map*) *pyramid*.

A 2-map pyramid [2] is a hierarchical data structure used for image analysis. Each level in a 2-map pyramid is a 2-map. The first level describes the initial full-resolution data; the other levels describe successive reductions of the previous levels. Usually, a pixel in the initial full-resolution 4-connected image is represented as a vertex in a 2D cubical complex, and adjacency relation between pixels is represented through edges in the complex. The reduction is obtained by applying operators that merge regions in the lower level into one region in the successive level (called *contraction operators*) and simplify the boundaries between the new merged regions (called *removal operators*). Each region in a coarser resolution image is a (connected) set of vertices, the representative of a region is an element of this set, called a *surviving vertex*, and other elements are called *non-surviving vertices*.

More formally, a 2-map  $(m+1)$ -level pyramid  $P$  is the set  $P = \{G^k\}_{0 \leq k \leq m}$  of 2-maps such that for each  $k$ ,  $0 < k \leq m$ ,  $G^k$  is obtained from  $G^{k-1}$  by contracting the cells (edges) in a set of cells  $C^{k-1}$  (contraction kernel) and removing the cells (edges) in a set of cells  $R^{k-1}$  (removal kernel). Several strategies have been proposed to choose the sets of the removed and contracted cells [8].

Another general multi-resolution framework, used mainly for simplicial complexes, called a *Multi-Complex*, has been introduced in [5].

## 4 Homology-Preserving Euler Operators

We review the Euler operators on cell complexes, proposed in [3], and we define the effect of homology-preserving Euler operators on the *IG* representing them.

### 4.1 Homology-Preserving Euler Operators on Cell Complexes

Operators that modify a cell complex, by modifying the number of cells in the complex and its Betti numbers, and maintain the validity of Euler-Poincaré

formula, are called *Euler operators*. In the literature, a variety of sets of basis Euler operators have been proposed, mainly for the 2D and the 3D case.

In [3], we have proposed a minimal set of Euler operators on cell complexes in arbitrary dimensions, which subsume all the other Euler operators proposed in the literature. These operators can be classified as:

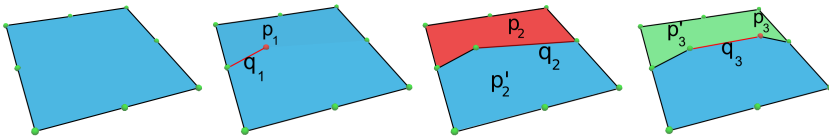
- homology-preserving operators:  $MiC(i + 1)C$  (*Make  $i$ -Cell and  $(i+1)$ -Cell*),
- homology-modifying operators:  $MiCiCycle$  (*Make  $i$ -Cell and  $i$ -Cycle*).

*Homology-preserving* operators  $MiC(i + 1)C$  change the number of cells in the complex  $\Gamma$ , by increasing the number  $n_i$  of  $i$ -cells and the number  $n_{i+1}$  of  $(i + 1)$ -cells by one. The Euler characteristic and the Betti numbers of the complex remain unchanged. Homology-preserving operator  $MiC(i + 1)C$  can create two new cells  $p$  and  $q$  from an existing  $i$ - or  $(i + 1)$ -cell, or insert the new cells in the complex.

The first type of  $MiC(i + 1)C$  operator has two instances. It either splits an existing  $i$ -cell  $p'$  in two by splitting its co-boundary, and creates an  $(i + 1)$ -cell  $q$  bounded by the two  $i$ -cells  $p$  and  $p'$ , or dually, it splits an existing  $(i + 1)$ -cell  $p'$  into two by splitting its boundary, and creates an  $i$ -cell  $q$  separating the two  $(i + 1)$ -cells  $p$  and  $p'$ . In both cases, the created  $i$ -cell appears exactly once on the boundary of the created  $(i + 1)$ -cell.

The second type of  $MiC(i + 1)C$  operator either creates an  $i$ -cell and an  $(i + 1)$ -cell bounded only by the  $i$ -cell, or dually, it creates an  $(i + 1)$ -cell and an  $i$ -cell bounding only the  $(i + 1)$ -cell. In both cases, the created  $i$ -cell appears exactly once on the boundary of the created  $(i + 1)$ -cell.

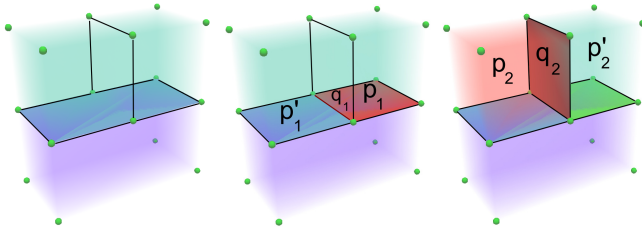
Figure 1 illustrates a sequence consisting of  $M0C1C(p_1, q_1)$  (second type, second instance),  $M1C2C(p_2, q_2)$  (first type, second instance) and  $M0C1C(p_3, q_3)$  (first type, first instance) in 2D. Figure 2 illustrates a sequence consisting of  $M1C2C(p_1, q_1)$  and  $M2C3C(p_2, q_2)$  (both of first type, second instance) in 3D. For brevity, we will consider only the operators of the first type.



**Fig. 1.** A sequence consisting of  $M0C1C$ ,  $M1C2C$  and  $M0C1C$  on a 2D cell complex;  $M0C1C$  creates 0-cell  $p_1$  and 1-cell  $q_1$ ,  $M1C2C$  creates 1-cell  $q_2$  and 2-cell  $p_2$ ,  $M0C1C$  creates 0-cell  $p_3$  and 1-cell  $q_3$

The inverse  $KiC(i + 1)C$  (*Kill  $i$ -Cell and  $(i+1)$ -Cell*) operators delete an  $i$ -cell and an  $(i + 1)$ -cell from  $\Gamma$ . The first type of  $KiC(i + 1)C$  operator is feasible in the following two cases:

- (i) the deleted  $(i + 1)$ -cell  $q$  is bounded by exactly two  $i$ -cells (the deleted  $i$ -cell  $p$  and the non-deleted  $i$ -cell  $p'$ ) and the deleted  $i$ -cell  $p$  appears exactly once on the boundary of  $(i + 1)$ -cell  $q$ ;



**Fig. 2.** A sequence consisting of  $M1C2C$  and  $M2C3C$  on a 3D cell complex;  $M1C2C$  creates 1-cell  $q_1$  and 2-cell  $p_1$ ,  $M2C3C$  creates 2-cell  $q_2$  and 3-cell  $p_2$

- (ii) the deleted  $i$ -cell  $q$  bounds exactly two  $(i + 1)$ -cells (the deleted  $(i + 1)$ -cell  $p$  and the non-deleted  $(i + 1)$ -cell  $p'$ ) and the deleted  $i$ -cell  $q$  appears exactly once on the boundary of  $(i + 1)$ -cell  $p$ .

In the first case, the effect of the operator is that the deleted  $i$ -cell  $p$  is replaced with the non-deleted  $i$ -cell  $p'$  in the boundary of each  $(i + 1)$ -cell  $r$  in the co-boundary of the deleted  $i$ -cell  $p$ . One copy of  $(i+1)$ -cell  $q$  is merged into  $(i+1)$ -cell  $r$  for each time  $i$ -cell  $p$  appears on the boundary of  $(i + 1)$ -cell  $r$ . The second case is dual.

*Homology-modifying* operators change both the number of cells in the complex  $\Gamma$  and its Betti numbers, and they change the Euler characteristic of  $\Gamma$ . They increase the number  $n_i$  of  $i$ -cells and the number  $b_i$  of non-bounding  $i$ -cycles by one. The inverse *KiCiCycle* (*Kill i-Cell and i-Cycle*) operators delete an  $i$ -cell and destroy an  $i$ -cycle, thus decreasing the numbers  $n_i$  and  $b_i$  by one.

### 4.2 Homology-Preserving Euler Operators on Incidence Graphs

$KiC(i + 1)C$  operator on an  $IG G = (N, A)$  deletes an  $i$ -node and an  $(i + 1)$ -node from  $N$ , and suitably reconnects the remaining nodes. Its first instance is feasible on  $IG G$  if

- $(i + 1)$ -node  $q$  is connected to exactly two different  $i$ -nodes  $p$  and  $p'$ , and
- there is exactly one arc in  $A$  connecting  $(i + 1)$ -node  $q$  and  $i$ -node  $p$ .

The effect of  $KiC(i + 1)C(p, q)$  on  $G$  is that

- nodes  $p$  and  $q$ , all the arcs incident in  $(i + 1)$ -node  $q$  and all the arcs incident in  $i$ -node  $p$  and connecting  $p$  to  $(i - 1)$ -nodes are deleted,
- all the arcs incident in  $i$ -node  $p$  and connecting  $p$  to  $(i + 1)$ -nodes are replaced with arcs connecting  $i$ -node  $p'$  to the same  $(i + 1)$ -nodes for each arc connecting  $i$ -node  $p$  to  $(i + 1)$ -node  $q$ .

In terms of the ordinary labeled incidence graph, let us denote as  $\varphi'(p', r)$  the label of the arc  $(p', r)$  after the simplification, where  $r$  is an  $(i+1)$ -node connected to the deleted  $i$ -node  $p$ . The label  $\varphi'(p', r)$  is increased by the product of the

label of the arc connecting nodes  $p'$  and  $q$  and the label of the arc connecting nodes  $p$  and  $r$  ( $\varphi'(p', r) = \varphi(p', r) + \varphi(p', q) \cdot \varphi(p, r)$ ).

The second instance of the  $KiC(i+1)C$  operator can be expressed as a modification of the  $IG G = (N, A)$  in a completely dual fashion.

The inverse  $MiC(i+1)C$  on an  $IG G = (N, A)$  also has two instances. The first instance is specified by the two inserted nodes ( $(i+1)$ -node  $q$  and  $i$ -node  $p$ ), the  $i$ -node  $p'$  that is the only  $i$ -node apart from  $i$ -node  $p$  that will be connected to  $(i+1)$ -node  $q$ , the  $(i+2)$ -nodes that will be connected to  $(i+1)$ -node  $q$ , and the  $(i-1)$ -nodes and  $(i+1)$ -nodes that will be connected to  $i$ -node  $p$ , together with the multiplicity (labels  $\varphi'$ ) of all the inserted arcs. It is feasible if all the specified nodes are in  $N$ , and the label  $\varphi(p', r)$  before the refinement for each  $(i+1)$ -node  $r$  that will be connected to  $i$ -node  $p$  is greater than or equal to  $\varphi'(p', q) \cdot \varphi(p, r)$ . Its effect is to add nodes  $p$  and  $q$  in  $N$  and all the specified arcs in  $A$  and to set  $\varphi'(p', r) = \varphi(p', r) - \varphi'(p', q) \cdot \varphi(p, r)$ . The second instance has a completely dual effect.

### 4.3 Homology-Preserving Operators on 2-Maps

Simplification operators have been defined on 2-maps in terms of elimination of darts from set  $D$  and modifications of permutations on the remaining darts. The simplification operators are called *removal* and *contraction*. They are the same as  $K0C1C$  and  $K1C2C$  operators, respectively.

## 5 Multi-resolution Model

We have defined and implemented a multi-resolution model for the topology of cell complexes represented through an  $IG$ , that we call a *Multi-Resolution Cell Complex (MCC)*. It is generated from the  $IG$  representing the cell complex at full resolution by iteratively applying  $KiC(i+1)C$  operators. The  $IG G_B = (N_B, A_B)$  obtained as a result of a specific simplification sequence (determined by the error criterion adopted) applied to the initial full-resolution graph is the coarsest representation of the topology of the complex, and we call it the *base graph*. It is the first ingredient of the multi-resolution model.

The second ingredient is the set  $\mathcal{M}$  of refinement operators, inverse to the simplification operators applied in the simplification process.

The third ingredient of the multi-resolution model is a dependency relation  $\mathcal{R}$  on the set  $\mathcal{M}$  plus  $\mu_0$ , where  $\mu_0$  is a dummy refinement that generates  $G_B = (N_B, A_B)$ . We define a dependency relation between refinements in  $\mathcal{M} \cup \mu_0$  as follows: refinement  $\mu$ , which introduces nodes  $p$  and  $q$ , *directly depends* on refinement  $\mu^*$  if and only if  $\mu^*$  creates at least one node that is connected to either  $p$  or  $q$  by  $\mu$ . The transitive closure of the direct dependency relation defined above is a partial order relation, since a node is never introduced twice by the refinements in  $\mathcal{M}$ . A multi-resolution model for the topology of cell complexes is, thus, a triple  $MCC = (G_B, \mathcal{M}, \mathcal{R})$ , where  $G_B$  is the  $IG$  representing the cell complex at the coarsest resolution,  $\mathcal{M}$  is the set of refinements inverse to



the simplifications applied in the generalization process, and  $\mathcal{R}$  is the direct dependency relation defined over  $\mathcal{M}$ .

The *MCC* can be encoded as a Directed Acyclic Graph (DAG), in which the root corresponds to modification  $\mu_0$ , i.e., to the creation of the base graph  $G_B$ , the other nodes correspond to the modifications in  $\mathcal{M}$ , and the arcs represent the direct dependency relation  $\mathcal{R}$ .

## 6 Selective Refinement

We discuss how to extract a large number of adaptive representations from an  $MCC = (G_B, \mathcal{M}, \mathcal{R})$  and briefly discuss some algorithmic aspects.

The set  $\mathcal{U} = \{\mu_0, \mu_1, \mu_2, \dots, \mu_m\} \subseteq \mathcal{M}$  of refinements in  $\mathcal{M}$  is *closed* with respect to dependency relation  $\mathcal{R}$  if for each  $1 \leq l \leq m$  in  $\mathcal{U}$ , each refinement on which refinement  $\mu_l$  depends is in  $\mathcal{U}$ . Let  $U = (\mu_0, \mu_1, \mu_2, \dots, \mu_m)$  be a sequence of the refinements belonging to  $\mathcal{U} \subseteq \mathcal{M}$ , such that, for each  $\mu_l \in U$  and each refinement  $\nu$  on which  $\mu_l$  depends,  $\nu = \mu_j \in U$ ,  $0 \leq j < l$ . Then,  $U$  is called a *feasible sequence*. The *front graph*  $G_U$  associated with a feasible sequence  $U$  is the graph obtained from the base graph  $G_B$  by applying the sequence of refinements  $U$ . It can be shown that any two feasible sequences  $U_1$  and  $U_2$  obtained from the same closed set  $\mathcal{U}$  produce the same front graph. Thus, a closed subset  $\mathcal{U}$  of refinements can be applied to the base *IG*  $G_B$  in any total order  $U$  that extends the partial order, producing an *IG*  $G_U$  at an intermediate resolution. If a feasible sequence  $U$  contains all refinements in  $\mathcal{M}$ , then the front graph  $G_U$  associated with  $U$  is the same as the *IG* at full resolution.

An *MCC* encodes the collection of all representations of a cell complex, at intermediate levels of resolution, which can be obtained from the base representation  $G_B$  by applying a closed set of modifications on  $G_B$ . From an *MCC* it is thus possible to dynamically extract representations of the topology of a cell  $n$ -complex at *uniform* and *variable* resolutions. The basic query for extracting a single-resolution representation from a multi-resolution model is known as *selective refinement*.

A selective refinement query on an *MCC* consists of extracting from it the *IG* with the minimum number of nodes, satisfying some application-dependent criterion. This criterion can be formalized by defining a Boolean function  $\tau$  over all nodes of an *MCC*, such that the value of  $\tau$  is *true* on nodes which satisfy the criterion, and *false* otherwise. An *IG*  $G = (N, A)$  is said to satisfy a criterion  $\tau$  if function  $\tau$  assumes the value *true* on all nodes in  $N$ . Thus, a selective refinement query consists of extracting from the *MCC* an intermediate graph of minimum size that satisfies  $\tau$ . Equivalently, it consists of extracting a minimal closed set  $\mathcal{U}$  of modifications from  $\mathcal{M}$  such that the corresponding complex satisfies  $\tau$ . Such closed set of modifications uniquely determines a front graph, which is obtained from the base graph  $G_B = (N_B, A_B)$  by applying to it all modifications from  $\mathcal{U}$  in any order that is consistent with the partial order defined by the dependency relation. The criterion  $\tau$  can be defined based on various conditions posed on the cells in the extracted complex, like the size of the cell (which may be expressed

as the maximum distance between its vertices or the diameter of its bounding box) or the portion of the complex in which the maximum resolution is required (while in the rest of the complex, the resolution may be arbitrarily low).

We have implemented a depth-first algorithm for the selective refinement query. The algorithm starts from the coarse  $IG G_B$  and recursively applies to it all refinements  $\mu_i$  which are required to satisfy the error criterion. In order that a new modification  $\mu$  be applied, all its ancestor modifications need to be applied before  $\mu$  to maintain the partial order. It can easily be proven that the result of a selective refinement algorithm is a graph  $G = G_U$  with minimal number of nodes among the graphs that can be extracted from the  $MCC$ , such that all nodes in  $G_U$  satisfy criterion  $\tau$ .

## 7 Experimental Results

The purpose of experiments is twofold. In the first set of experiments, we have tested two simplification strategies to build the  $MCC$ : one approach is based on performing simplifications one at the time, and the other on performing a set of independent simplifications. In the second set, we show the capabilities of the  $MCC$  to extract adaptive representations at variable resolutions, and compare timings for the two approaches. We have performed the experiments on 2D and 3D simplicial complexes available on the Web and encoded in an  $IG$ , that become cell complexes after undergoing some simplification. The initial size of these complexes is between 400K and 953K triangles for 2D data sets, and between 68K and 577K tetrahedra for 3D data sets. Experiments have been performed on a desktop computer with a 3.2Ghz processor and 16Gb of memory.

To build the  $MCC$ , we start from the  $IG$  at full resolution and perform all the feasible simplifications in the order guided by some criterion  $\tau$  until the coarsest representation is reached. The implementation of the simplification algorithm is independent of criterion  $\tau$ . We have used a geometric criterion computed on the vertices of the deleted cells, and we have implemented two different simplification approaches. In the first one, called *step-by-step simplification*, simplifications are extracted from the priority queue in ascending order and performed if still feasible. After each simplification, the local connectivity of the nodes involved in it changes and each new feasible simplification is enqueued. The algorithm ends when there are no more feasible simplifications.

The second approach, called *batch simplification*, tries to execute at each step a large number of feasible independent simplifications (that involve nodes not involved by any other already selected simplification). At each step, we build a priority queue with all the feasible simplifications sorted in ascending order. We select a set of simplifications from the queue, we perform all of them, and we initialize a new priority queue.

In Table 1 we summarize the results obtained with the two approaches. The columns show, from left to right, data set name (*Data set*), total number of cells (*Cells*), number of simplifications (*Simpl. Num.*), time needed to perform them (*Simpl. Time*), time needed to build the  $MCC$  (*MCC Time*), storage

cost of the *MCC* (*MCC storage*), time needed to perform all the refinements in the *MCC* (*Ref. Time*), storage cost of the cell complex at full resolution (*Full complex*) and storage cost of the base complex (*Base complex*).

**Table 1.** Experimental results for the DAG construction. The storage cost is expressed in Megabytes and the computation time in seconds.

Data set	Cells	Simpl. Num.	Simpl. Time	MCC Time	MCC storage	Ref Time	Full complex	Base complex
Step-by-step simplification								
<i>Eros</i>	2859566	1429781	74.4	5.3	254.9	18.1	349.0	0.0002
<i>Hand</i>	1287532	643694	35.4	2.3	117.2	7.58	157.1	0.01
2D <i>VaseLion</i>	1200002	599999	26.7	2.1	105.8	6.8	146.4	0.00028
Batch simplification								
<i>Eros</i>	2859566	1429781	218.8	6.4	241.0	18.7	349	0.0002
<i>Hand</i>	1287532	643741	99	2.6	120.7	7.6	157.1	0.004
<i>VaseLion</i>	1200002	599999	90.7	2.3	110.5	7.7	146.4	0.00028
Step-by-step simplification								
<i>VisMale</i>	297901	147594	45.1	0.6	40.4	5.1	48	0.46
<i>Bonsai</i>	1008357	498790	380.6	2.7	146.9	27.2	162.5	1.8
3D <i>Hydrogen</i>	2523927	1248743	8643.8	7.8	395.7	419.5	407.4	4.4
Batch simplification								
<i>VisMale</i>	297901	148116	69.2	0.7	37.6	2.5	48	0.28
<i>Bonsai</i>	1008357	501524	305.8	2.69	126.4	10.4	162.5	0.89
<i>Hydrogen</i>	2523927	1253913	1412.9	7.4	321.3	33.9	407.4	2.7

We can notice that the time needed to perform all the refinements is always much less than the time needed to perform all the simplifications (refinement is 5 to 10 times faster than simplification). An important aspect is that the storage cost of the *MCC* structure plus the base graph is less than the storage cost of the graph at full resolution, with the exception of the largest tested (*Hydrogen*) data set using the step-by-step method. Although the total number of simplifications is slightly higher for the batch simplification approach, the time required to perform all simplifications that lead to the base complex is less in the case of step-by-step simplification, since it requires fewer computations. On the other hand, the *MCC* generated through batch simplification uses less memory and consequently can be visited in less time. We have observed that

**Table 2.** Experimental timing results (in seconds) for extraction at variable resolution

2D				3D			
Data set	Perc.	Refinement step-by-step	Time batch	Data set	Perc.	Refinement step-by-step	Time batch
<i>Eros</i>	50%	0.80	0.92	<i>VisMale</i>	50%	3.45	0.12
	80%	1.42	1.01		80%	3.77	0.15
	100%	2.63	2.60		100%	4.01	0.53
<i>Hand</i>	50%	0.31	0.57	<i>Bonsai</i>	50%	15.3	0.65
	80%	0.45	0.65		80%	17.4	0.69
	100%	1.20	1.19		100%	19.1	1.88
<i>VaseLion</i>	50%	0.73	0.69	<i>Hydrogen</i>	50%	106.3	8.1
	80%	1.01	0.99		80%	127.7	8.7
	100%	1.10	1.06		100%	172.1	11.3

the DAGs produced by the batch simplification have less dependency relations compared to the ones produced by step-by-step simplification.

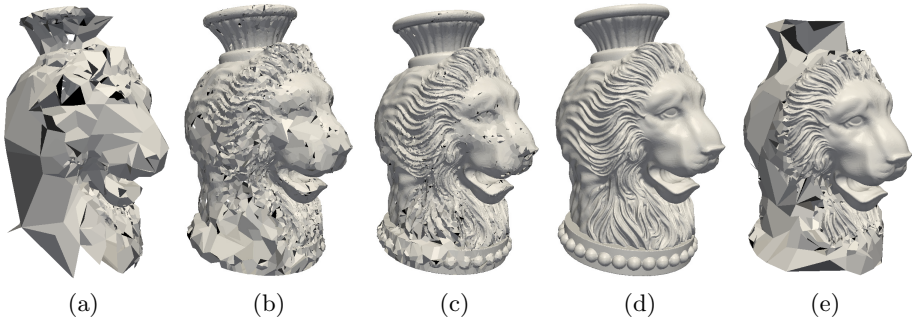
In Table 2, we show timing results for performing extractions at variable resolution. Column *Perc.* indicates the desired percentage of operations performed inside a query box. *Refinement Time* indicates the time needed to perform the required number of refinements with the step-by-step method (*step-by-step*) or the batch (*batch*) simplification methods. The query box has been chosen by hand to cover a relevant part for each data set and with size between 15 and 30 percent of the whole data set at full resolution. We can observe that the extraction times for refinements are similar for the two methods in the 2D case, while they differ considerably in the 3D case. Note that in 2D each 1-node in the incidence graph is connected with at most two different 0-nodes and two different 2-nodes, while in 3D there is a variable number of arcs between 1-nodes and 2-nodes: a larger number of arcs in the *IG* leads to a larger number of dependency relations in the *MCC*. This has a significant impact in the use of a simplification method that reduces the *MCC* complexity.

In Figure 3, we show examples of refinement queries at uniform and variable resolution performed on the *VaseLion* data set. The holes that seem to appear in the crown of the lion are rendering artifacts.

## 8 Discussion and Outlook

We compare the 2D instance of the *MCC* defined on *IGs* with the pyramid model defined on 2-maps.

The first advantage of the *MCC* over pyramidal models is its space efficiency. This is a consequence of the fact that the *IG* occupies less memory than the *n*-map representing the same complex. Each dart in an *n*-map corresponds to a path in the *IG* representing the same *n*-dimensional cell complex from an



**Fig. 3.** In (a), (b) and (c) the representations obtained from the *MCC* after 10000, 50000 and 2000000 refinements, respectively. In (d), the complex at full resolution of the *VaseLion* data set. In (e) the representation obtained with a query at variable resolution.

$n$ -node to a 0-node. For each dart, the set of  $n$  involutions is encoded plus a pointer for each entity which points to the geometric and attribute description of such entities, as discussed in [10]. This leads to a storage cost of  $B * (2n + 1)$  items, where  $B$  is the number of darts. For  $n = 2$ , it can be easily seen that  $B = 4n_1$ , where  $n_1$  denotes the number of 1-cells in the complex, while the number of arcs in the *IG* is equal to  $4n_1$  (they can be encoded through  $8n_1$  pointers), and the number of nodes is equal to the total number of cells in the complex. In general, we can observe that each path in the *IG* is defined by a set of  $n - 1$  arcs, and the storage cost is less than  $B * (n - 1)$  items, since the paths overlap. We have evaluated on a set of 2-complexes and 3-complexes, the ratio between the storage cost for the *IG* and for the  $n$ -map; the value for this ratio is around 50% for 2-complexes, and around 18% for 3-complexes.

The second advantage is a wider representation domain. *IGs* can represent arbitrary cell complexes, while  $n$ -maps can represent (closed orientable) quasi-manifolds, which are a class of pseudo-manifolds.

We plan to apply the homology-preserving operators to the computation of homology of a cell complex. An arbitrary cell complex at full resolution can be simplified by applying a sequence of homology-preserving operators, until no further simplification is possible. Homology can be computed on the simplified complex using standard techniques [1]. Homology generators on the simplified complex can be computed using the method similar to the ones in [12,4], proposed for images and complexes represented as  $n$ -G-maps, respectively, and propagated from such complex to the full-resolution complex using the *MCC*.

**Acknowledgments.** This work has been partially supported by the Italian Ministry of Education and Research under the PRIN 2009 program, and by the National Science Foundation under grant number IIS-1116747.

## References

1. Agoston, M.K.: Computer Graphics and Geometric Modeling: Mathematics. Springer-Verlag London Ltd. (2005) ISBN:1-85233-817-2
2. Brun, L., Kropatsch, W.G.: Introduction to Combinatorial Pyramids. In: Bertrand, G., Imiya, A., Klette, R. (eds.) Digital and Image Geometry. LNCS, vol. 2243, pp. 108–128. Springer, Heidelberg (2001)
3. Čomić, L., De Floriani, L.: Topological Operators on Cell Complexes in Arbitrary Dimensions. In: Ferri, M., Frosini, P., Landi, C., Cerri, A., Di Fabio, B. (eds.) CTIC 2012. LNCS, vol. 7309, pp. 98–107. Springer, Heidelberg (2012)
4. Damiand, G., Gonzalez-Diaz, R., Peltier, S.: Removal Operations in  $nD$  Generalized Maps for Efficient Homology Computation. In: Ferri, M., Frosini, P., Landi, C., Cerri, A., Di Fabio, B. (eds.) CTIC 2012. LNCS, vol. 7309, pp. 20–29. Springer, Heidelberg (2012)
5. De Floriani, L., Magillo, P., Puppo, E.: Multiresolution Representation of Shapes Based on Cell Complexes. In: Bertrand, G., Couprie, M., Perrotton, L. (eds.) DGCI 1999. LNCS, vol. 1568, pp. 3–18. Springer, Heidelberg (1999)
6. Edelsbrunner, H.: Algorithms in Combinatorial Geometry. Springer, Berlin (1987)
7. Edmonds, J.: A Combinatorial Representation for Polyhedral Surfaces. Notices Amer. Math. Soc. 7, 646 (1960)
8. Haxhimusa, Y., Glantz, R., Kropatsch, W.G.: Constructing Stochastic Pyramids by MIDES - Maximal Independent Directed Edge Set. In: Hancock, E.R., Vento, M. (eds.) GbRPR 2003. LNCS, vol. 2726, pp. 24–34. Springer, Heidelberg (2003)
9. Kovalevsky, V.: Axiomatic Digital Topology. Journal of Mathematical Imaging and Vision 26(1-2), 41–58 (2006)
10. Lévy, B., Mallet, J.L.: Cellular Modelling in Arbitrary Dimension Using Generalized Maps. Technical report, INRIA (1999)
11. Lienhardt, P.: N-Dimensional Generalized Combinatorial Maps and Cellular Quasi-Manifolds. International Journal of Computational Geometry and Applications 4(3), 275–324 (1994)
12. Peltier, S., Ion, A., Haxhimusa, Y., Kropatsch, W.G., Damiand, G.: Computing Homology Group Generators of Images Using Irregular Graph Pyramids. In: Escolano, F., Vento, M. (eds.) GbRPR 2007. LNCS, vol. 4538, pp. 283–294. Springer, Heidelberg (2007)

# Convergence of Level-Wise Convolution Differential Estimators

Damien Gonzalez<sup>1</sup>, Rémy Malgouyres<sup>1</sup>, Henri-Alex Esbelin<sup>1</sup>,  
and Chafik Samir<sup>2</sup>

<sup>1</sup> Clermont-Université, LIMOS UMR 6158 CNRS,  
Complexe des Cézeaux, 63172 Aubière, France  
{damien.gonzalez,remy.malgouyres}@u-clermont1.fr,  
Alex.Esbelin@univ-bpclermont.fr

<sup>2</sup> Clermont-Université, ISIT UMR 6284 CNRS,  
Complexe des Cézeaux, 63172 Aubière, France  
chafik.samir@u-clermont1.fr

**Abstract.** Differentials estimation of discrete signals is almost mandatory in digital segmentation. In our previous work, we introduced the fast level-wise convolution (LWC) and its complexity of  $O(2n \cdot \log 2(m))$ . We present convergence proofs of two LWC compatible kernel families. The first one is the pseudo-binomial family, and the second one the pseudo-Gaussian family. In the experimental part, we compare our method to the Digital Straight Segment tangent estimator. Tests are done on different digitized objects at different discretization step using the DGtal library.

**Keywords:** Differential estimator, discrete differential operator, fast convolution, sparse differential operator, FFT.

## Introduction

Digital segmentation algorithms such as active contour models often use signal parameters as energy. Estimation of differentials is almost mandatory for most of them as they use regularization terms like the snake algorithm [7]. Previous works are divided into two categories: the non convolutional and the convolutional methods.

1. Non convolutional methods. The Digital Straight Segment (DSS) tangent estimator [8,2] extracts maximal DSS and computes their tangents. One of the advantages of this method is its ability to detect corners, its convergence rate is  $O(\frac{1}{3})$ . The Taylor polynomial approximation [12] fits the values of a digital function by a polynomial. It introduces a roughness parameter to relax the function values within an interval. It has a bounded maximal error of  $O(h^{\frac{1}{1+k}})$  for the  $k^{th}$  derivative and a resolution  $h$ , its convergence rate is  $O(\frac{1}{1+k})$ . The Global min-curvature estimator (GMC) is a curvature estimator along digital contours. It first estimates the uncertainty of tangents using a tangential cover, and then minimizes the global curvature.

2. Convolutional methods. The binomial convolution Method (BC) [11,3] approximates differentials with finite differences after applying a digital version of the scale space [9] function smoothing, using integer-only binomial coefficients as a convolution mask. It is noise resistant, has a convergence rate of  $O(h^{\frac{2}{3}k})$  and a complexity of  $O(n.m)$  with  $n$  the size of the image and  $m$  the one of the convolution mask.

In our previous work [4] we defined a differential estimator based on a Level-wise Convolution Kernel, thus obtaining a LWC method. Compared to BC, it has a better complexity and speed and appears to provide similar error results in experiments. In this paper, we use two techniques to generate level-wise kernels from a given smoothing kernel. We then focus on theorems and proofs of convergence for two kernel families and experimental convergence confirmation as well. More precisely, under some relaxed assumptions of absence of noise and for floating point methods, we prove that LWC converges in  $O(h^2)$  with both a level-wise pseudo-Gaussian kernel and a level-wise binomial kernel. In the presence of a uniform noise in  $O(h^\alpha)$ , we prove a result similar to [3] for the faster LWC method. We have not been able to prove such a result for a pseudo-Gaussian kernel or a level-wise pseudo-gaussian kernel. At last, to further improve comparison with other methods we also show results comparing the precision and speed of our LWC with respect to DSS. The scope of this paper being limited to first order derivative estimators we don't compare here our method with GMC.

Section 1 gives definitions of the LWC and two compatible kernels. One based on the Gaussian function and one based on the binomial coefficients. In the following two sections we present mathematical proofs of convergence for the LWC using the two kernels. First the level wise pseudo-binomial kernel ( $LWB_n$ ) and then the level-wise pseudo-Gaussian kernel LWG. In the experimental results section, we show comparisons with DSS in terms of precision and runtime.

## 1 Level-Wise Convolution

When dealing with derivatives estimation, one of the most classical methods is to use the finite differences  $(f(x+h) - f(x))/h$ . Although effective in continuous geometry, it cannot be applied as such to discrete images because derivative values would be limited to integers. A solution is to average each pixel of the image with its neighbour, a process called smoothing. This mathematical operation is known as the convolution product of a function in the integers interval  $f : [0, n] \rightarrow [0, n]$ , the image to be convolved and a function  $H : [0, n] \rightarrow [0, n]$ , the averaging kernel. Gaussian function as a kernel is the standard in this field as described by Lindeberg in the scale space theory [9]. The resulting image can then serve to compute differentials, using finite differences with a convolution by a differential operator  $\Delta$  as the kernel. Figure 1, is an example of the convolution of a digital function 1, 2, 2, 4, 5 by a binomial coefficients kernel 1, 2, 1. To preserve the image scale, each value has to be divided by the mass (or weight) of the kernel (the sum of all its values) in this case  $W = 2^n = 4$ . After we convolve the smoothed image by a differential operator to obtain the derivatives (first



order). There are three kinds of first order derivative operator, the centered one, as in the example  $f(1) - f(-1)$ , the backward difference  $(f(0) - f(-1))$  and the forward difference  $(f(1) - f(0))$ . The choice of the derivative operator depends on symmetric properties of the smoothing kernel. For a centered kernel (odd size) we use the centered operator. For an even size kernel we can convolve it with left or right shift (we use the opposite smoothing kernel shift). Out of the range of the discretization there is no information on the function values. The convolution loses precision when those pixels are required. This is known as the border problem and for the example of Figure 1, unknown values are set to 0. In the experimental part we only use functions for which we know those values. The major drawback of this method in the discrete paradigm is its complexity of  $O(n.m)$ , with  $n$  being the size of the image and  $m$  the size of the kernel.

**Definition 1.** *The discrete convolution product (noted  $*$ ) is a transform of two discrete functions  $F : \mathbb{Z} \rightarrow \mathbb{Z}$  and  $H : \mathbb{Z} \rightarrow \mathbb{Z}$ . At least one is of finite support.*

$$(F * H)(x) = \sum_{i \in \mathbb{Z}} f(x - i).H(i)$$

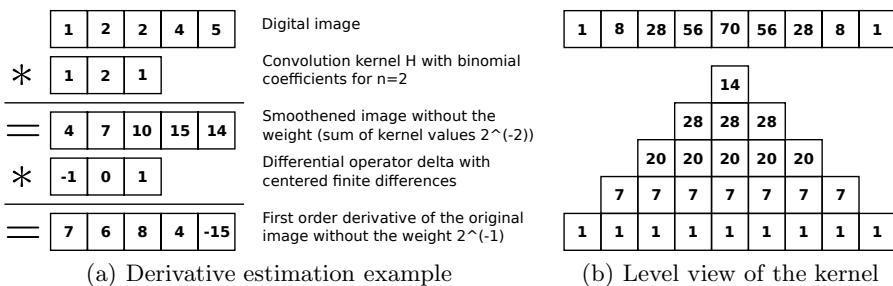
$H$  is said to be a smoothing kernel when  $\sum_{i \in \mathbb{Z}} H(i) = 1$ .

Definition 1 shows the discrete convolution product of an image  $f$  with a kernel  $H$  for the pixel  $x$  of  $f$ .

$$(F * H)(n) = \sum_{j=0}^k \sum_{i=-k+j}^{k-j} f(n + i).(H(-k + j) - H(-k + j - 1)) \quad (1)$$

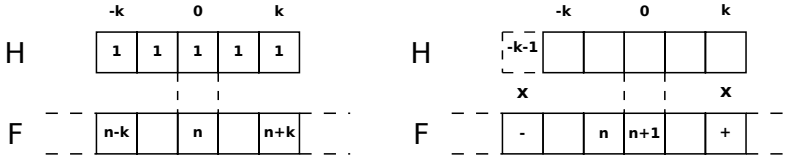
$$(F * H)(n + 1) = (F * H)(n) - H(-k).F(n - k - 1) + H(k).F(n + k) \quad (2)$$

Looking at the right part of Figure 1 the kernel can be viewed in a multilevel way. The values are the same for the whole level and by adding them all we



**Fig. 1.** (a). First the detailed process of convolving a digital image with a binomial coefficients kernel, second the convolution of the smoothed image with the central differential operator of central finite differences. (b). Level view of the binomial kernel for  $\binom{n}{k}$  with  $n = 8$  and  $k$  the line index.

obtain the original kernel as shown in the upper right part. The discrete convolution product can be rewritten to apply one level of the kernel at a time, as shown in Equation (1), the convolution of the function  $f$  with the kernel  $H$  only for the pixel  $n$ . The first loop  $\sum_{j=0}^k$  iterates through all levels and the second one  $\sum_{i=-k+j}^{k-j}$  through the whole current level. The complexity has changed to  $O(\sum_{i=0}^{\lfloor m/2 \rfloor} 2i + 1)$  since  $m = 2k + 1$ , but Equation (1) only concerns one pixel and since each level of the kernel has the same value, we only need to convolve for the first pixel as indicated in Equation (2) and Figure 2.



**Fig. 2.** Convolution of image  $f$  with kernel  $H$  of size  $m = 2k + 1$  centered in 0

The left part is the convolution of the pixel  $n$  and the right part is the convolution of pixel  $n + 1$  using the previous result. Since each level of the kernel has the same value Figure 1, we only need to subtract the product of  $H(-k)$  and  $F(n - k - 1)$  and to add the product  $H(k)$  and  $F(n + k)$  to the convolution of pixel  $F(n)$  to get the result.

### 1.1 Complexity

The resulting complexity depends on the number of levels of the kernel and we only use kernels with  $\log_2$  bounded number of levels. We have a  $O(2n \cdot \log_2(m))$  complexity which is smaller than the binomials convolution of  $O(n \cdot m)$  in [11,3] and theoretically slightly better to the latest complexity of the FTT of

$O(\frac{39}{4} N \cdot \log_2(N))$  in [5,10,6]. This complexity is only for the FFT, and in order to compute a convolution using Fourier transform, several steps are required. The first step, is to apply the transform to the image to be in Fourier space. The second is to multiply the image by the Gaussian kernel. The third is to apply an inverse transform to the result of the previous multiplication to return to the original space. Using other bounds for the number of levels than the logarithm can increase or decrease the complexity and it could be interesting to have a kernel with a fixed number of levels. In higher dimension the complexity will be  $O(n^{dim} \cdot \log_2(m))$  with  $dim$  being the dimension.

### 1.2 Kernel Compatibility and Extension to Higher Dimensions

For a kernel to be compatible with this method (central difference), it must have an even size to avoid data shift. It has a  $\log_2$  bounded number of levels in order to have the complexity described in Subsection 1.1. The convolution should work in higher dimensions using the tensor product of one dimension kernels. The use of  $n$  dimension kernels is possible if they are separable in 1 dimension elements.

## 2 Kernels Presentation

We introduce two discrete kernels of low complexity. They are level-wise versions of two known kernels, the binomial kernel and the Gaussian kernel. There are two methods to generate level-wise kernels. The first one consists in averaging kernel's coefficients per level (pseudo-binomial). The second uses a logarithmic function (pseudo-Gaussian). The fastest discrete converging method for the first three order derivative is the binomial coefficient kernel. And the fastest in terms of computational times is the Gaussian kernel. Kernels are symmetrical:  $H(x) = H(-x)$  and they decrease from center to periphery:  $H(x) > H(y)$  for  $|x| < |y|$ . First let us introduce classical binomial and Gaussian kernels.

**Definition 2.** *The binomial kernel is a discrete approximation of the Gaussian function. Its coefficients are obtained using the binomials  $\binom{n}{k}$ , with  $n$  the width of the kernel and  $k$  its index.  $B_n : \mathbb{Z} \rightarrow \mathbb{Z}$ .*

$$B_n(k) = \binom{2n}{n - k}$$

Let us recall that the classical Gaussian kernel used in scale space is  $g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$ . We introduce now the smoothing kernel obtained by discretizing this one.

**Definition 3.** *The Gaussian kernel's coefficients are the ones of the centered Gaussian function.  $G_\lambda \mathbb{Z} \rightarrow \mathbb{R}$ .*

$$G_\lambda(i) = \alpha 2^{-\lambda i^2} \quad \text{where } \alpha = \sum_{i \in \mathbb{Z}} 2^{-\lambda i^2}$$

The parameter  $\alpha$  is chosen to get 1 as total weight. The parameter  $\lambda$  determines the length of the mask.

### 2.1 Pseudo-gaussian Kernel

To create this kernel, we start from the continuous Gauss formula. We create a rough kernel with its level number bounded by the  $\log_2$  function  $LWG_{\lambda\gamma}$  Definition 4. It is always centered in 0 with  $\alpha$  representing the weight such as the integral of the kernel is equal to 1 in order not to scale the image after convolution. Parameter  $\gamma$  controls the number and length of levels.

**Definition 4.** *The level-wise pseudo-Gaussian (LWG).  $LWG_{\lambda,\gamma} : \mathbb{Z} \rightarrow \mathbb{R}$ .*

$$LWG_{\lambda,\gamma}(i) = \alpha 2^{-\lambda \gamma^{2 \cdot \lfloor \frac{\log_2(|i|)}{\log_2 \gamma} \rfloor}}$$

### 2.2 Building Complexity

Since we can not predict at which value of  $i$  the level change will occur, we have to compute the values for the whole kernel size  $O(m)$  with  $m$  being the size of the kernel. The building process can be speeded up by using the left arithmetic shift to compute powers of 2.

### 2.3 Pseudo-binomial Kernel

We used the binomial coefficients as a basis for this kernel since it is a good discretization of the Gaussian function  $G$ . It is a level-wise kernel and the values of the levels are the sum of the binomial coefficients between two boundaries controlled by the floor function. In Definition 5, we have the pseudo-binomial kernel  $LWB_n$ . The integers parameters are:  $m$  is the size of the kernel and  $n$  the number of levels. Let us denote  $s(i)$  is the signature of the  $i$ . Figure 3 illustrates the relation between the binomial kernel and  $BLW$ . On top  $B_n$  is represented by the different levels which size are increasing by power of two.  $BLW$  level values are the average of the values in the corresponding level.

**Definition 5.** *The level-wise pseudo-binomial ( $LWB_n$ ).  $LWB_n : \mathbb{Z} \rightarrow \mathbb{Z}$ .*

$$LWB_n(0) = \frac{1}{2^{2^{\alpha+1}-2}} \binom{2^{\alpha+1} - 2}{2^{\alpha} - 1} \quad \text{for } n \neq 0$$

$$LWB_n(i) = \frac{1}{2^{2^{\alpha+1}-2 + \lfloor \log_2(|i|) \rfloor}} \left( \sum_{k=2^{\lfloor \log_2(|i|) \rfloor}}^{k=2^{1+\lfloor \log_2(|i|) \rfloor} - 1} \binom{2^{\alpha+1} - 2}{2^{\alpha} - 1 + s(i)k} \right)$$

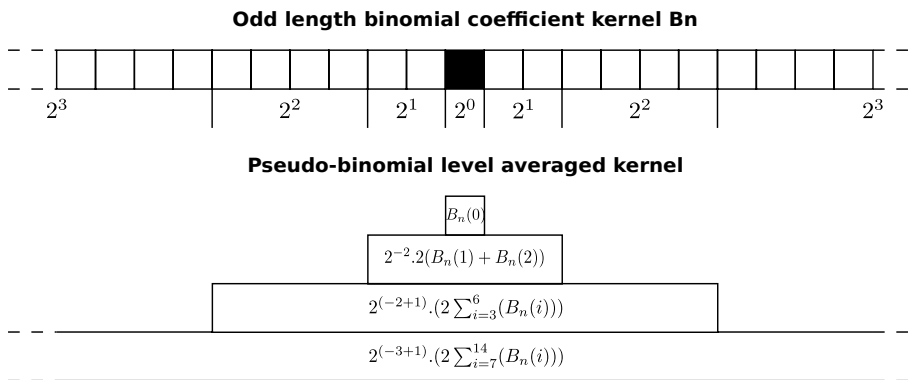
$$\text{where } \begin{cases} s(i) = +1 & \text{if } n > 0 \\ s(i) = -1 & \text{if } n < 0 \end{cases} \quad \text{and } m = 2^{\alpha+1} - 2$$

### 2.4 Building Complexity

To minimize the complexity we use the Pascal triangle building method to compute the binomial coefficients. For the memory management, we use the upper bound  $\frac{4^n}{8n\sqrt{\pi n}}$  to allocate our triangle's line.

## 3 Convergence

In this section, we prove convergence results for discretization of functions  $f : \mathbb{R} \rightarrow \mathbb{R}$  having a bounded third continuous derivative  $f^{(3)}$ . At the end, we compare the convergence rate of our method to the ones in literature.



**Fig. 3.** Top. Odd length binomial kernel  $B_n$  with the center marked by the black square. Bottom. Pseudo-binomial kernel  $LWB_n$ . The lower line represent power of 2 coefficients determining the level size.

### 3.1 Convergence and Error Estimation for Unnoisy Images

We prove here a convergence result for discretization by real numbers of functions having a bounded third continuous derivative. Such a result is convenient for implementations using floating numbers. It is valid for all the kernels we mentioned in the previous section.

For a discretization step  $h$ , let  $\Gamma : \mathbb{Z} \rightarrow \mathbb{R}$  be a real discretization of  $f : \mathbb{R} \rightarrow \mathbb{R}$  and  $f'$  the first order derivative of  $f$ .

$$h\Gamma(i) = f(ih)$$

It is convenient for implementations using floating numbers. The following theorem is worthwhile for all the kernels we mentioned in the previous section.

**Theorem 1.** *Let  $H$  be any non negative symmetric smoothing kernel and  $\Delta * H$  be the associated derivating kernel, where  $\Delta$  is the central difference operator defined by  $\Delta(-1) = \frac{1}{2}$  and  $\Delta(+1) = -\frac{1}{2}$  and  $\Delta(i) = 0$  for other values of  $i$ . Let  $x_0 = i_0h$  with  $i_0 \in \mathbb{Z}$ . Suppose that  $\sum_{j \geq 1} j^2 H(j)$  exists. Suppose moreover that  $H(i)$  is decreasing for non negative  $i$ .*

(1) (local convergence):  $\lim_{h \rightarrow 0} ((\Delta * H) * \Gamma)(x_0) = f'(x_0)$

(2) (rate of local convergence):

$$((\Delta * H) * \Gamma)(x_0) - f'(x_0) \sim_{h \rightarrow 0} \frac{1 + 3 \sum_{j \geq 1} j^2 H(j)}{6} f^{(3)}(x_0) h^2$$

(3) (uniform convergence):

$$\text{Sup} \{ |(\Delta * H) * \Gamma(x) - f'(x)| ; x \in h\mathbb{Z} \} \leq \frac{1 + 3 \sum_{j \geq 1} j^2 H(j)}{6} \|f^{(3)}\|_\infty |h|^2$$

*Proof.* Let  $a_i = (\Delta * H)(i)$ . We have to evaluate  $\frac{1}{h} \left( \sum_{i \in \mathbb{Z}} a_i f(x + ih) \right) - f'(x)$

For each  $i$  in  $\mathbb{Z}$ , there are  $x'_i$  between  $\text{Min}\{x, x + (i - 1)h\}$  and  $\text{Max}\{x, x + (i - 1)h\}$  such that

$$f(x + ih) = f(x) + ihf'(x) + \frac{(ih)^2}{2} f^{(2)}(x) + \frac{(ih)^3}{6} f^{(3)}(x'_i)$$

Now it is easy to check that  $\sum_{i \in \mathbb{Z}} a_i = 0$  and  $\sum_{i \in \mathbb{Z}} ia_i = 1$  and  $\sum_{i \in \mathbb{Z}} i^2 a_i = 0$  and

$\sum_{i \in \mathbb{Z}} i^3 a_i$  exists ; hence we have:

$$\frac{1}{h} \left( \sum_{k \in \mathbb{Z}} a_k f(x + ih) \right) - f'(x) = \frac{h^2}{6} \left( \sum_{i \in \mathbb{Z}} i^3 a_i f^{(3)}(x_i) \right)$$

But now, as the  $i^3 a_i$  are non negative, we have  $\sum_{i \in \mathbb{Z}} |i^3 a_i| = \sum_{i \in \mathbb{Z}} i^3 a_i = 2 \sum_{i \geq 1} i^3 a_i =$

$\sum_{i \geq 1} i^3 (H(i - 1) - H(i + 1)) = 1 + 3 \sum_{j \geq 1} j^2 H(j)$  and the two first results are coming

immediately when  $h$  tends to 0.

Noticing that  $\left( \sum_{i \in \mathbb{Z}} i^3 a_i f^{(3)}(x_i) \right) \leq \|f^{(3)}\|_\infty \sum_{i \in \mathbb{Z}} |i^3 a_i|$ , the reader would immediately consider that the proof is complete.

### 3.2 Error Estimation for Noisy Images

We consider here the following model for noisy images: let  $f : \mathbb{R} \rightarrow \mathbb{R}$  be the real image; suppose that the sequence  $\Gamma : \mathbb{Z} \rightarrow \mathbb{Z}$  is a noisy discretization with a uniformly bounded error  $|h\Gamma(i) - f(hi)| \leq Kh^\alpha$ , where  $\alpha \in [\frac{1}{2}, 1]$ ,  $K \in R_+^*$  and  $h \in R_+^*$ .

**Theorem 2.** *Suppose that  $f$  is a  $C^3$  function and  $f^{(3)}$  is bounded. If  $m = \lfloor h^{2(\alpha-3)/5} \rfloor$ , then we have  $|(LWB_{2m} * \Gamma)(n) - f'(nh)| \in O(h^{(4\alpha-2)/5})$  for sufficiently large  $h$ .*

*Proof.* We prove the following inequality in a standard way:

$$|LWB_{2m} * \Gamma(n) - f'(nh)| \leq \frac{2 + 3m^2}{6} h^2 \|\phi^{(3)}\|_\infty + \frac{2Kh^{\alpha-1}}{4m} \binom{2m}{m}$$

From the well known Stirling's formula,  $n \sim \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$  we get for  $m \rightarrow +\infty$

$$\frac{1}{4^m} \binom{2m}{m} \sim \frac{1}{4^m} \frac{\sqrt{4\pi m} \left(\frac{2m}{e}\right)^{2m}}{\left(\sqrt{2\pi m} \left(\frac{m}{e}\right)^m\right)^2} = \frac{\sqrt{4\pi m}}{2\pi m} = \frac{1}{\sqrt{\pi m}}$$

Hence choosing  $m = \lfloor h^{\frac{2(\alpha-3)}{3}} \rfloor$  provides the result.

Notice that this result is significantly worse than the corresponding result for Binomial smoothing kernel (non level-wise) which has a convergence proof for C2 functions. However this is a preliminary result that could probably be strengthened. It seems possible to generalize the results to C2 functions in the case of the pseudo-binomial kernel. Otherwise, we fail to prove such a result for pseudo-Gaussian (even non level-wise) smoothing kernel. Ulterior work may show counter examples for the pseudo-Gaussian kernel in terms of convergence or convergence rate for C2 functions.

### 3.3 State of the Art

Maximal error of differential estimation mainly depends on the image. This is the reason why we include the image in our convergence rate in Figure 4. We can not bound the error for an arbitrary family of functions since our upper bound depends on the norm of the derivatives, but we can bound the error for a set of functions. Trigonometrical functions have the following bound  $\|f'''(x)\|_\infty = 1$ . For polynomial functions, the bound will depend on the degree of the function.

BC	DSS	Taylor P.	LWC
1/3	2/3	1/2	$\ f'''(x)\  \cdot \frac{h^2}{6} \sum (i^3 ai)$

**Fig. 4.** Convergence rates of different estimators for the first order derivative. Datas were taken from [12].

## 4 Experiments

We have created level-wise version of two kernel families. The Gaussian kernel used in the scale space theory (SCT), and the binomial kernel used in discrete geometry. With level-wise convolution we have reduced the convolution complexity. Figure 9 and Figure 6 show experiment results of computational time verifying theoretical complexity. Figure 7 and Figure 8 show experimental convergence rates for the two families of kernel on a ball and on an ellipsis.

The tests of the two kernels have been done on 2D digitized images generated by the DGtal library [1]. The convolution is a 1D kernel applied on the contour of the images, so computational time does not suffer from the 1D tests presented in [4]. Since our convergence proofs implies that objects are C3, we choose the ones that have this property. The y axis is the Euclidian norm of the

Discretization step	0.1	0.01	0.001
Pseudo-Gaussian $\gamma$	5	20	80
Pseudo-Gaussian $\lambda$	1	1	1
Pseudo-binomial $\alpha$	4	8	12

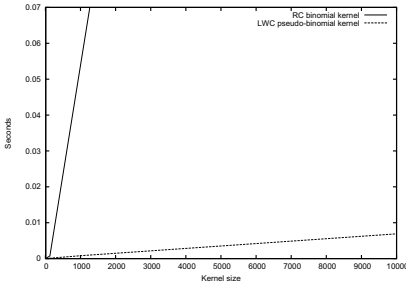
**Fig. 5.** Parameters of kernels families used in the experimental part

difference between the expected value and the estimators value. The x axis is the logarithm of the inverse of the discretization step. The LWG used in this experiments LWG\_exp Definition 6 is a derivated form of the one presented in Definition 4.

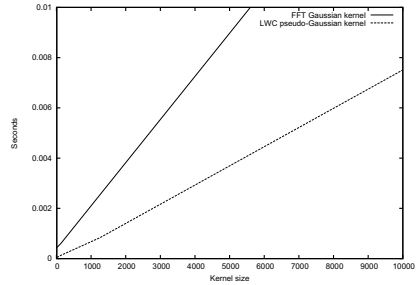
**Definition 6.** Experiments version of the LWG kernel,  $LWG\_exp_{\lambda,\gamma} : \mathbb{Z} \rightarrow \mathbb{R}$ .  $\log_2 e$  is a result from the change of  $e$  in 2.

$$LWG\_exp_{\lambda,\gamma} = \alpha 2^{-\lambda \log_2 e \cdot \gamma^{2^{\lfloor \frac{\log_2(|i|)}{\log_2 \gamma} \rfloor}}}$$

Kernels parameters were chosen empirically, but they are the same for each function as shown in Figure 5.



(a) BC vs LWC pseudo-binomial

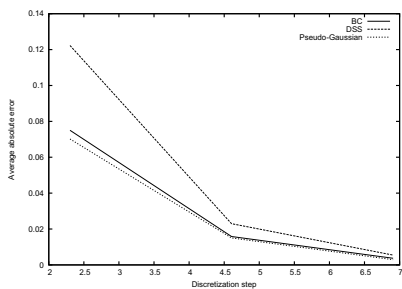


(b) FFT vs LWC pseudo-Gaussian

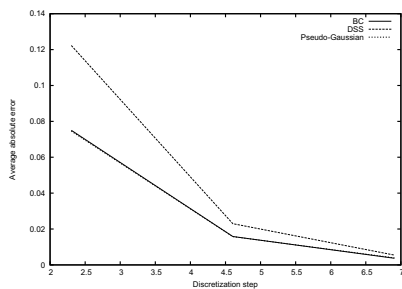
**Fig. 6.** Computational time comparison of convolution of an image of size 1000 by kernels of the same size. (a). Binomial kernel using regular convolution and pseudo-binomial using LWC. (b). Gaussian kernel using FFT and pseudo-Gaussian kernel using LWC.

The computational time results of Figure 6 were obtained with a digitized ball of radius 1. Both methods are not optimized for computational time result. DSS is implemented using integers and our method uses floating-point numbers. The difference is significant enough, and this regardless of implementations to conclude that our method is faster.



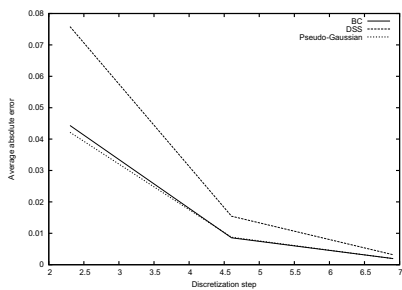


(a) Pseudo-binomial kernel family

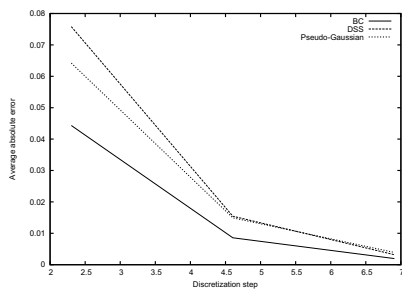


(b) Pseudo-Gaussian kernel family

**Fig. 7.** Experimental convergence rates on a digitized ball of radius 1. (a). Comparison results for the pseudo-binomial kernel family. (b). Comparison results for the pseudo-Gaussian kernel family

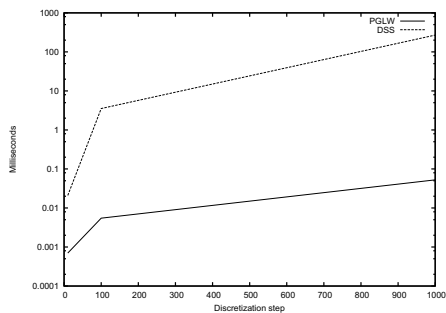


(a) Pseudo-binomial kernel family



(b) Pseudo-Gaussian kernel family

**Fig. 8.** Experimental convergence rates on a digitized ellipsis of large radius 1. (a). Comparison results for the pseudo-binomial kernel family. (b). Comparison results for the pseudo-Gaussian kernel family.



**Fig. 9.** Computational times in milliseconds on a logarithmic scale as a function of the resolution

## 5 Conclusion

We have presented a uniform convergence proof for LWC for two compatible kernel families. We have experimentally confirmed convergence as well. The experimental part shows the precision even with empiric parameters on selected differentiable objects. Also LWC is significantly faster than DSS (as implemented in the DGtal library). It is important to note that both implementations of DSS and LWC that we used may not be optimized. In our future work we will focus on parameters selection and the adaptivity of our kernel using multi-pass convolutions for higher order differentials. Noise resistance proof for pseudo-Gaussian kernels will be further investigated. We will also do some tests in higher dimension and higher derivative order. The last goal will be the GPU implementation allowing the multi-pass approach to improve the runtime.

**Acknowledgement.** The research leading to these results has received funding from the KIDICO project of the French *Agence Nationale de la Recherche* (Grant Agreement ANR-2010-BLAN-0205-02).

## References

1. DGtal: Digital geometry tools and algorithms library, <http://iris.cnrs.fr/dgtal>
2. De Vieilleville, F., Lachaud, J.O.: Comparison and improvement of tangent estimators on digital curves. *Pattern Recognition* 42(8), 1693–1707 (2009)
3. Esbelin, H., Malgouyres, R., Cartade, C.: Convergence of binomial-based derivative estimation for 2 noisy discretized curves. *Theoretical Computer Science* 412(36), 4805 (2011)
4. Gonzalez, D., Malgouyres, R., Esbelin, H.A., Samir, C.: Fast Level-Wise Convolution. In: Barneva, R.P., Brimkov, V.E., Aggarwal, J.K. (eds.) *IWCIA 2012*. LNCS, vol. 7655, pp. 223–233. Springer, Heidelberg (2012), [http://dx.doi.org/10.1007/978-3-642-34732-0\\_17](http://dx.doi.org/10.1007/978-3-642-34732-0_17)
5. Heideman, M., Johnson, D., Burrus, C.: Gauss and the history of the fast fourier transform. *IEEE ASSP Magazine* 1(4), 14–21 (1984)
6. Johnson, S., Frigo, M.: A modified split-radix fft with fewer arithmetic operations. *IEEE Transactions on Signal Processing* 55(1), 111–119 (2007)
7. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. *International Journal of Computer Vision* 1(4), 321–331 (1988)
8. Lachaud, J., Vialard, A., de Vieilleville, F.: Fast, accurate and convergent tangent estimation on digital contours. *Image and Vision Computing* 25(10), 1572–1587 (2007)
9. Lindeberg, T.: *Scale-space theory in computer vision*. Springer (1994)
10. Lundy, T., Van Buskirk, J.: A new matrix approach to real ffts and convolutions of length  $2^k$ . *Computing* 80(1), 23–45 (2007)
11. 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)
12. Provot, L., Gérard, Y.: Estimation of the Derivatives of a Digital Function with a Convergent Bounded Error. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) *DGCI 2011*. LNCS, vol. 6607, pp. 284–295. Springer, Heidelberg (2011)

# Concurrency Relations between Digital Planes

Peter Veelaert, Maarten Slembrouck, and Dirk Van Haerenborgh

Ghent University,

Valentin Vaerwyckweg, B9000 Ghent, Belgium

{peter.veelaert,maarten.slembrouck,dirk.vanhaerenborgh}@ugent.be

**Abstract.** In this paper we examine concurrency relations between planes whose position is not precisely known. The simplest case consists of four planes, where we have to determine whether the four planes can be forced to pass through one common intersection point by moving them slightly within specified limits. We prove that if such a concurrency relation is possible then it can be found in a finite number of steps by a simple geometrical construction. This result remains valid for larger collections of planes, with multiple concurrency relations, provided each pair of relations shares at most one plane, and the relations do not form cycles.

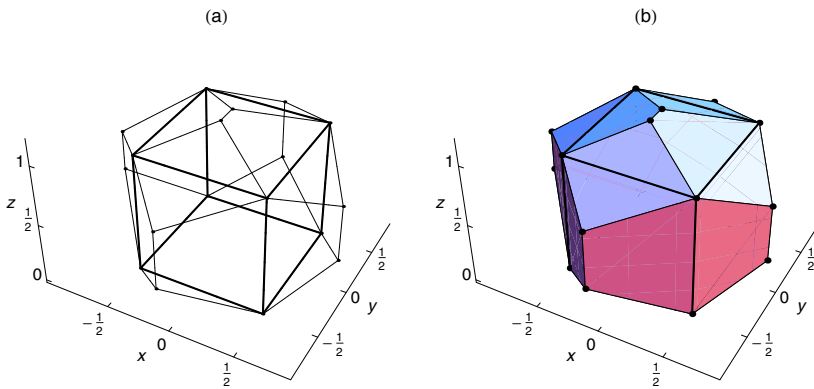
## 1 Introduction

In 3D concurrencies of planes occur when at least 4 planes meet at a common point. However, when the planes are extracted from real data, e.g., point clouds, even when four planes are concurrent in the real scene, the extracted planes will not meet at a common point, because of imprecise data. Concurrencies may still be possible, but within prespecified tolerances for the positions and slopes of the planes. Our goal is to find hidden concurrencies by simple geometric constructions that are easy to implement. A geometric construction here means that the concurrent planes pass through lines and points constructed from intersections and spans of a finite set of initial points. When the initial points have integral coordinates all the computations can be done with rational numbers, with the significant advantage that the results are exact.

In this paper the tolerances on the position of a plane are defined by a convex polytope, which we will call a domain. Thus given a set of  $n$  domains and a set of  $m$  concurrency relations, our goal is to determine whether the concurrency relations can hold for  $n$  plane positions chosen from the domains.

Part of this work is motivated by the renewed interest in visual hulls [1]. Although known for some time [2, 3], multi-camera systems now provide means to acquire detailed visual hulls in real time. One former obstacle for acquiring visual hulls was that they need accurate silhouettes. Although obtaining reliable silhouettes from image segmentation remains difficult [4], motion detection algorithms with sophisticated background models are now able to extract silhouettes that are sufficiently reliable [5, 6]. Each camera added to the network improves the accuracy of the hull. In fact, an abundance of cameras allows us to deal with occlusions and missing segments.

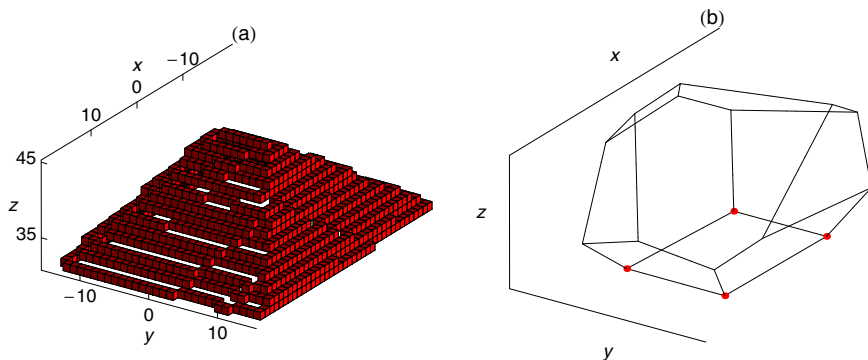
Fig. 1 shows a synthetic image of the visual hull of a cubic box when viewed by 4 cameras. The visual hull is a polytope which results from the intersection of 4 cones, and the halfspace defined by the ground plane. Each cone is generated by the light rays that pass through the projection center of one of the cameras and the silhouette of the box as seen by this camera. The precise combinatorial structure of the visual hull depends on the position of the box relative to the cameras. Thus the number of vertices, edges, and facets may vary when a box is displaced. However, at a corner of the box, typically the visual hull will have multiple supporting planes that meet at a common point. For example, in Fig. 1, six planes meet at the corner in front. Clearly, if we want to embed a box inside a visual hull, it is interesting to know where  $n$  planes, with  $n > 3$ , meet at a common point. Fig. 2 shows the real data for a rectangular box. Because of noisy data, concurrencies that should occur at corners are not present, but slight displacements of the facets can make them reappear.



**Fig. 1.** Synthetic generation of a visual hull. (a) shows the edges of the box (bold lines) together with the edges of its visual hull. (b) shows the same result but with facets shown as filled polygons. Because of the edges and facets are quite small, a slight change of the camera positions may already give a different combinatorial structure.

Tolerances for planes and lines also arise naturally in digital geometry [7–9]. The results proven in this paper closely resemble previous results on the concurrency of lines [10]. As before, the domain of a plane [11], i.e., the set of all parameters that fall within the prespecified tolerances, plays an important role, since all proofs are given in the dual parameter space.

Section 2 introduces concurrency relations in the primal as well as in the dual space. In Section 3 we clarify what it means for a concurrency relation to be constructible. The paper is concluded in Section 4.



**Fig. 2.** Visual hull of a box acquired by 4 cameras. (a) Shows a voxel representation of a close up of the upper part of the visual hull. Because each horizontal layer of the visual hull is constructed separately, the surface contains gaps between voxels. (b) Shows the result of segmentation, and then fitting planes to each segment. The dots indicate the vertices that rest on the ground plane. The combinatorial structure depends strongly on the fitting process. For example,  $L_\infty$  fitting gives a different outcome than  $L_2$  fitting.

## 2 Concurrency Relations

### 2.1 Concurrencies in Primal Space

A convenient way to model the tolerance of a plane is to describe its parameters by a convex polytope, also called the domain. There are several ways to arrive to such a domain. Let  $W$  be a finite set of points in  $\mathbb{R}^3$ , and let  $\tau > 0$  be a positive thickness. Then the set of all positions the plane can take such that it passes within a distance  $\tau/2$  of  $W$  is defined as the set of parameter points  $(a, b, c)$  that satisfy

$$-\frac{\tau}{2} \leq z_i - ax_i - by_i - c \leq \frac{\tau}{2}, \quad \forall (x_i, y_i, z_i) \in W.$$

The thickness  $\tau$  can be chosen as large as needed to accommodate for the uncertainty of the point positions. Alternatively, the tolerance of a plane can be defined by the way it separates point sets. Let  $U$  and  $V$  be finite and non-empty subsets of  $\mathbb{Z}^3$  such that  $U$  can be linearly separated from  $V$ . Then each parameter  $(a, b, c)$  point that satisfies

$$\begin{aligned} z_i - ax_i - by_i - c &> 0, & \forall (x_i, y_i, z_i) \in U \\ z_j - ax_j - by_j - c &< 0, & \forall (x_j, y_j, z_j) \in V. \end{aligned} \tag{1}$$

corresponds to a plane that separates the two subsets. We will define the domain  $D(U, V)$  as the set of parameter points  $(a, b, c)$  that satisfy

$$\begin{aligned} z_i - ax_i - by_i - c &\geq 0, & \forall (x_i, y_i, z_i) \in U \\ z_j - ax_j - by_j - c &\leq 0, & \forall (x_j, y_j, z_j) \in V. \end{aligned}$$

Thus  $D(U, V)$  is the topological closure of the set defined by (1). In general a domain is a convex polyhedron. If the inequalities are such that the domain is bounded, it is a polytope. A domain becomes infinite if the position of one of the separating planes is orthogonal to the  $xy$ -plane. To avoid this complication, we will assume that there exists at least one plane in space to which none of the separating planes are orthogonal. Then, we can always rotate the coordinate axes such that all domains will be bounded convex polytopes, and all planes can be defined by an equation of the form  $z - ax - by - c = 0$ .

We will allow special kinds of domains. A domain will be a planar segment (or polygon) in  $\mathbb{R}^3$  if we require that the plane that separates two point sets also has to pass through a given point  $p_1$ . Similarly, a domain will be a line segment if the separating plane has to pass through two points, and trivially, a domain is a single point if we require that the separating plane has to pass through three given points. These special cases will be useful when we want to verify some results in extreme situations.

Each domain  $D$  defines a set of possible plane positions defined by the equation  $z = ax + by + c$ , where  $(a, b, c) \in D$ . To define concurrencies, the plane at position  $(a, b, c)$  will be denoted as  $\Pi_{(a,b,c)}$ . If we denote with  $P_{conc}(a_1, \dots, c_4)$  the determinant

$$\begin{vmatrix} 1 & a_1 & b_1 & c_1 \\ 1 & a_2 & b_2 & c_2 \\ 1 & a_3 & b_3 & c_3 \\ 1 & a_4 & b_4 & c_4 \end{vmatrix},$$

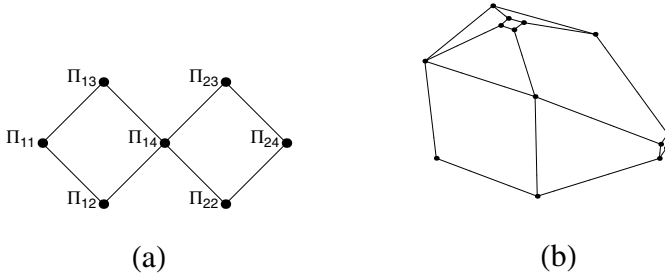
then the four planes  $\Pi_{(a_1,b_1,c_1)}, \dots, \Pi_{(a_4,b_4,c_4)}$  are **concurrent** if  $P_{conc}(a_1, \dots, c_4) = 0$ .

This polynomial will also be denoted as  $P_{conc}(\Pi_1, \Pi_2, \Pi_3, \Pi_4)$ .

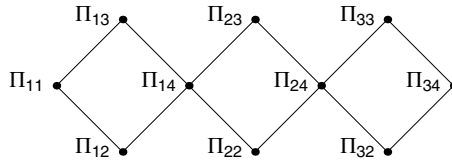
Our main problem is now the following. Suppose we are given  $n$  domains  $D_i$  and  $m$  geometric relations  $P_j$ . Can we find a plane configuration  $\Pi_{(a_1,b_1,c_1)}, \dots, \Pi_{(a_n,b_n,c_n)}$ , with  $(a_i, b_i, c_i) \in D_i$  such that  $P_j(a_i, \dots, b_k) = 0$  holds for all  $m$  relations  $P_j$ ?

This is a non-linear programming problem with linear inequalities for the domains and non-linear equations for the concurrency relations. Hence, we may expect that in general a solution can only be found by solving systems of non-linear equations. We shall show, however, that for certain configurations a solution can be found by a geometric construction. The feasibility of a geometric solution depends on how the concurrencies are linked to each other. An example of the configurations we envision is shown in Fig. 3, as a graphical diagram. Each node represents a plane, each quadrangle represents a concurrency relation. Thus Fig. 3 represents the concurrency relations  $P_{conc}(\Pi_{11}, \Pi_{12}, \Pi_{13}, \Pi_{14})$  and  $P_{conc}(\Pi_{14}, \Pi_{22}, \Pi_{23}, \Pi_{24})$ , sharing one plane,  $\Pi_{14}$ .

The results we prove are valid for all chains of this type. Chains can be of arbitrary length, but two subsequent relations cannot share not more than one plane. Fig. 4 shows a chain with 3 relations. Obviously, we can also have multiple chains with no planes in common between chains. In this case each chain is handled separately.



**Fig. 3.** (a) shows a chain with two concurrency relations,  $P_{conc}(\Pi_{11}, \Pi_{12}, \Pi_{13}, \Pi_{14})$  and  $P_{conc}(\Pi_{14}, \Pi_{22}, \Pi_{23}, \Pi_{24})$  share one plane  $\Pi_{14}$ . (b) shows how such a chain can arise in the primal space. A polytope has 2 small facets, each adjacent to 4 large facets (some facets are hidden). By changing the position of the large facets slightly, the small facets could disappear and be replaced by two vertices. Each of the five large facets in (b) corresponds to a plane  $\Pi_{ij}$  in (a). The plane  $\Pi_{14}$  corresponds to the large facet incident to both small facets.



**Fig. 4.** A chain for three concurrency relations. A chain with  $m$  relations will comprise  $3m + 1$  planes. Planes are labeled such that plane  $\Pi_{ij}$  is involved in the  $i$ -th relation. Planes that are involved in two relations receive labels of the form  $\Pi_{i4}$ .

### 2.2 Concurrency in Dual Space

We will call the  $xyz$ -space the primal space, and the parameter space the dual space. Geometrically,  $n$  planes  $\Pi_{(a_i, b_i, c_i)}$  in the primal space are concurrent if and only if their parameters  $(a_i, b_i, c_i)$  are coplanar in the dual space. As a result,  $n$  parameter domains will give rise to  $n$  concurrent planes if we can find  $n$  coplanar parameter points, one point from each domain. Clearly, this is equivalent to the assertion that there is at least one plane in the parameter space that crosses all the domains. Such a plane is called a common stabbing. Thus, finding concurrencies in the primal space, is equivalent to finding common stabbings in the dual space. Concurrency relations for lines correspond to line stabbings [12, 13], while concurrency relations for planes correspond to plane stabbings. When  $n$  parameter points, where  $n \geq 4$  lie on a stabbing plane in the dual space, the corresponding  $n$  planes in the primal space are concurrent. Both conditions are expressed by the same algebraic equation,  $P_{conc} = 0$ . Furthermore, stabbing planes in the dual space correspond directly to intersection points in

the primal space. If the stabbing plane has the form  $b = \gamma - \alpha a - \beta c$ , the common intersection point in the primal space is  $(\alpha, \beta, \gamma)$ .

For example, for the concurrency relations of Fig. 3, we must find two common stabbings in the dual space, one for the domains  $D_{11}, D_{12}, D_{13}, D_{14}$ , and a second for  $D_{14}, D_{22}, D_{23}, D_{24}$ . What complicates the stabbing problem, however, is that the intersection line of the two stabbing planes must cross  $D_{14}$ . This is not guaranteed by the mere existence of two separate stabbings, which means that the concurrency problem cannot be solved locally.

### 3 Constructible Solutions

Our goal is to show that solutions can be found by geometrical constructions that start from known points. These known points may either be the points used to define the domains, i.e., points in  $U$  and  $V$ , but even when  $D$  is given without specifying  $U$  and  $V$ , we can always indicate a set of known points in the primal space.

**Definition 1.** Let  $D$  be a convex, polytopal domain of plane parameters with vertex set  $V_D$ . Let  $\{(a_i, b_i, c_i), \dots\}$  be a subset of vertices in  $V_D$  that lie on a common facet  $F$ . Then the point  $(x, y, z)$  that lies at the intersection of the sets defined by

$$z = a_i x + b_i y + c_i, \quad \forall (a_i, b_i, c_i) \in (F \cap V_D)$$

is called a *support point* of  $D$ .

Thus, each facet of the polytope defines a unique support point in the primal space. This support point is defined as the common intersection point for the different positions a plane can take as defined by one facet of its domain. It is also easy to see that the support points of  $U$  and  $V$  are sufficient to define  $D(U, V)$ . In fact, if  $S$  denotes the set of all support points of  $D(U, V)$ , then  $D(U, V) = D(U \cap S, V \cap S)$ .

We will make extensive use of the dualism between the primal and dual space. A vertex  $v = (a, b, c)$  in the dual space represents the plane  $\Pi$  at the position defined by the equation  $z = ax + by + c$ . We will denote this relationship as  $v' = \Pi$ , or vice-versa  $\Pi' = v$ . Similarly, the stabbing plane  $\Gamma$  defined as  $c = \gamma - \alpha a - \beta b$ , with parameters  $\alpha, \beta, \gamma$  corresponds to the primal point  $p = (\alpha, \beta, \gamma)$ . The dual plane  $\Gamma$  contains all the parameters of primal planes that pass through the primal point  $p$ . We will denote this relationship as  $p = \Gamma'$ , or vice-versa  $p' = \Gamma$ . A vertex always lies on at least 3 facets. Hence, a vertex  $v$  of a domain is the dual of a plane  $v' = \Pi$  that contains at least three support points. We will therefore call this primal plane  $v'$  a support plane. A support plane  $v'$  is the affine span of a subset of the support points.

#### 3.1 Meet and Join Operations in the Primal Space

Suppose we are given a connected chain of  $m$  concurrency relations and  $n = 3m + 1$  domains. Our goal is to show that if the concurrency relations can be satisfied



by  $n$  planes, one plane chosen from each domain, then it is always possible to construct geometrically a solution from the support points. From the support points we construct new points and planes by applying either the meet  $\wedge$  or the join operator  $\vee$ . Let  $\Pi_1 \wedge \Pi_2 \wedge \dots$  denote the intersection of the planes  $\Pi_1, \Pi_2, \dots$ . Let  $p_1 \vee p_2 \vee \dots$  denote the affine span of the points  $p_1, p_2, \dots$ . The affine span of three points is a plane, provided the points are in general position. The intersections of two planes is a line, provided the planes are not parallel.

A possible construction for the problem of Fig. 3 may now take the form:

$$\begin{aligned}
 \Pi_{11} &= p_{11} \vee q_{11} \vee s_{11} \\
 \Pi_{12} &= p_{12} \vee q_{12} \vee s_{12} \\
 \Pi_{13} &= p_{13} \vee q_{13} \vee s_{13} \\
 \Pi_{14} &= p_{14} \vee q_{14} \vee (\Pi_{11} \wedge \Pi_{12} \wedge \Pi_{13}) \\
 \Pi_{22} &= p_{22} \vee q_{22} \vee s_{22} \\
 \Pi_{23} &= p_{23} \vee q_{23} \vee s_{23} \\
 \Pi_{24} &= p_{24} \vee q_{24} \vee (\Pi_{14} \wedge \Pi_{22} \wedge \Pi_{23}),
 \end{aligned} \tag{2}$$

where  $p_{ij}, q_{ij}, s_{ij}$  are support points, and the indices point out the corresponding domain, i.e.  $p_{11}$  is a support point of  $D_{11}$ . However, many more constructions of this form are possible, depending on which support points are selected, and in what order. Our goal is to show that if the concurrency problem has a solution, then there always exists a solution that can be constructed geometrically by meet and join operations on the support points and previously constructed points and planes. Since there are only a finite number of possible constructions, this means that the existence of a solution can be demonstrated or disproved in a finite number of steps.

In a construction like this it is important that all the planes are defined, and that the concurrency relations are automatically satisfied when the points are in general position. For example, the same construction would be invalid if the fourth plane was constructed as  $\Pi_{14} = p_{14} \vee q_{14} \vee s_{14}$ , because then in general the planes  $\Pi_{11}, \Pi_{12}, \Pi_{13}, \Pi_{14}$  would not be concurrent.

### 3.2 Meet and Join Operations in the Dual Space

The meet and join operations of (2) provide one simple example of a possible construction. The constructions that we will need can be much more complicated. In fact, the concurrencies will first be established in the dual space, and then translated into the primal space. In the dual space concurrencies are established by constructing stabbing planes by meet and join operations that start from the vertices of the domains. For example, for the configuration of Fig. 3 we may have a stabbing plane of the form

$$\Gamma_1 = v_{11} \vee v_{12} \vee v_{13},$$

where  $v_{ij}$  denotes a vertex of domain  $D_{ij}$ . But we also allow more complicated expressions of the form

$$\Gamma_2 = (\Gamma_1 \wedge (v_{14a} \vee v_{14b})) \vee v_{22} \vee v_{23}$$

where  $\Gamma_2$  is constructed from vertices of domains involved in the second relation and a previously constructed stabbing plane  $\Gamma_1$  from the first relation.

### 3.3 Constructible Stabbings Yield Constructible Primal Planes

Expressions in the dual space can always be translated into expressions that involve support points in the primal space. In fact, each vertex in the dual space corresponds to a support plane in the primal space, which is the affine span of three support points. For example, if the points  $p_{1j}, q_{1j}, r_{1j}$  lie on the facet that corresponds to  $v_{1j}$ , then

$$v_{1j} = (p_{1j} \vee q_{1j} \vee r_{1j})'$$

and we have

$$\Gamma_1 = \bigvee_{j=1,\dots,3} (p_{1j} \vee q_{1j} \vee r_{1j})',$$

which specifies how the stabbing plane can be constructed by taking duals from affine spans of support points.

However, the construction of the stabbing planes in terms of support points still leaves some freedom with respect to the parameters of the primal planes  $\Pi_{ij}$ . Suppose that we have constructed  $m$  stabbing planes from the domain vertices such that the intersection line  $\Gamma_n \wedge \Gamma_{n+1}$  crosses the domain  $D_{n4}$  for each subsequent pair. Then we still have to make clear how we can derive the parameters of the planes  $\Pi_{ij}$  that satisfy the concurrency relations. To this end we will select in each domain a parameter point that lies on the stabbing plane, but where necessary we introduce additional constraints, which are explained below, to remove remaining degrees of freedom.

The additional constraints are chosen as simple as possible, e.g., by requiring that a parameter point lies on an edge of a domain. We have to make a distinction between domains that are involved either in one or in two relations. First, for a domain  $D_{ij}$  involved in one relation, if the stabbing plane  $\Gamma_i$  passes through one or more vertices of  $D_{ij}$ , we select one of these vertices. If  $\Gamma_i$  contains no vertex, the stabbing plane still passes through at least one edge of domain  $D_{ij}$ . In this case we select the intersection point of the edge and the stabbing plane. Second, for a domain  $D_{i4}$  that is involved in two relations, we select the intersection point of a facet of  $D_{i4}$  and the intersection line  $\Gamma_i \wedge \Gamma_{i+1}$ . Clearly, all the selected parameter points are constructible in the dual space, since they lie either at a vertex, at the intersection of an edge and a constructible stabbing plane, or at the intersection of a facet and two constructible stabbing planes. For example,  $(\Gamma_1 \wedge \Gamma_2 \wedge (u_{14} \vee v_{14} \vee w_{14}))'$  represents a plane in the primal space whose parameter point is defined in the dual space as the intersection of two stabbing planes and a facet spanned by the vertices  $u_{14}, v_{14}, w_{14}$ . In turn, each vertex can be written as the dual of the affine span of three support points, e.g.,  $u_{14} = (p_{14} \vee q_{14} \vee s_{14})'$ . Hence, when a parameter point  $(a, b, c)$  is constructible in the dual space from the vertices, the corresponding plane  $\Pi_{(a,b,c)}$  is constructible in the primal space from the support points.

### 3.4 Proof of the Main Result

We start with a simple result for one concurrency relation with 4 primal planes.

**Proposition 1.** *If four domains can be stabbed by a common dual plane, then there is a dual plane passing through 3 domain vertices that stabs the fourth domain.*

*Proof.* The requested stabbing is easily found by three rotations. Let  $a_{11}, a_{12}, a_{13}, a_{14}$  denote four arbitrarily chosen points on the common stabbing plane, one point chosen from each domain, i.e.,  $a_{ij} \in D_{ij}$ . We let the stabbing plane rotate around the axis  $a_{13}a_{14}$  until we hit a vertex in one of the two other domains. Without loss of generality we assume that this vertex lies in  $D_{11}$  and call it  $v_{11}$ . Next we let the plane rotate around the axis  $a_{14}v_{11}$  until we hit a second vertex either in  $D_{12}$  or  $D_{13}$ . Without loss of generality we may assume this vertex lies in  $D_{12}$ , and call it  $v_{12}$ . Finally, we let the plane rotate around  $v_{11}v_{12}$ , until we hit a third vertex either in  $D_{13}$  or  $D_{14}$ . The resulting plane passes through 3 vertices in 3 different domains and still stabs the fourth domain.  $\square$

Note that a domain vertex corresponds to a supporting plane. Thus the above result translates into: there exist three supporting planes, such that there is a fourth separating plane that passes through the common intersection point of the three supporting planes and two support points of the fourth domain.

To prove the main theorem, we will extend the previous result. However, in order to ensure that the stabbings stay within the common domains  $D_{i4}$ , we introduce visual hulls in the dual, which should not be confused with the visual hulls of Section 1. Suppose we have a convex polytope  $D$ , and two vertices  $v_1$  and  $v_2$  not in  $D$ . We define a visual hull of  $D$  as seen from the line  $v_1 \vee v_2$  as follows:

$$H(v_1 \vee v_2, D) = \{p \in \mathbb{R}^3 : (p \vee v_1 \vee v_2) \cap D \neq \emptyset\}. \tag{3}$$

In other words the visual hull is the set of all points that lie on planes that pass through  $v_1 \vee v_2$  and cross  $D$ . Note that this a variant of the more common definition of a visual hull. In our case the light rays are not lines but planes, and the projection center is not a point but a line. It is also clear that the intersection of a visual hull  $H(v_1 \vee v_2, D)$  with another domain  $D'$  is again a polytope, whose vertices can be constructed by meet and join operations on  $v_1, v_2$ , the vertices of  $D$ , and the vertices of  $D'$ .

We will now prove the general theorem. The proof is given in the dual space, but it is helpful to keep the dualities of Table 1 in mind. The dual space column in Table 1 indicates how the primal plane can be constructed from domain vertices and duals of stabbing planes.

**Theorem 1.** *Suppose we have a connected chain of  $m$  concurrency relations, with  $3n + 1$  domains  $D_{11}, \dots, D_{m4}$ . If there exist  $3n + 1$  primal planes  $\Phi_{11}, \dots, \Phi_{m4}$ , with  $\Phi'_{ij} \in D_{ij}$ , that satisfy the concurrency relations, then there is a second set of primal planes  $\Pi_{11}, \dots, \Pi_{m4}$  with  $\Pi'_{ij} \in D_{ij}$ , where each  $\Pi_{ij}$  can be constructed by meet and join operations on the support points and previously constructed planes.*

**Table 1.** Dualities between primal and dual space

dual space	primal space
$\Gamma_i$ passes through vertex $v$	support plane spanned by 3 support points of $v'$
$\Gamma_i$ crosses edge $v_1v_2$	plane spanned by 2 support points in $v'_1 \cap v'_2$ and by $\Gamma'_i$
line $\Gamma_i \wedge \Gamma_{i+1}$ crosses facet	plane spanned by 1 support point, $\Gamma'_i$ and $\Gamma'_{i+1}$

*Proof.* The general idea of the proof is to start from stabbing planes in the dual space that correspond to a solution, which is known to exist, and then rotate the stabbing planes until they pass through a sufficient number of vertices, edges or facets of the domains.

**First Stage.** We start with the first relation in the chain. Since there is a solution, there is also stabbing plane  $\Gamma_1$  in the dual space that stabs the 4 domains of the first relation. Let  $a_{11}, a_{12}, a_{13}$  denote three arbitrarily chosen points on the common stabbing plane, one point chosen from each domain  $D_{11}, D_{12}, D_{13}$ . Let  $a_{14}$  be a point chosen from  $D_{14}$  on the line  $\Gamma_1 \cap \Gamma_2$ . We let the stabbing plane  $\Gamma_1$  rotate around the axis  $a_{13} \vee a_{14}$  until we hit a vertex in one of the two other domains  $D_{11}$  or  $D_{12}$ . Without loss of generality we assume that this is a vertex of  $D_{11}$ , which we call  $v_{11}$ . Next we let the plane rotate around the axis  $a_{14} \vee v_{11}$ , until we hit a second vertex either in  $D_{12}$  or  $D_{13}$ . Assume this vertex lies in  $D_{12}$ , and call it  $v_{12}$ .

We now have a stabbing plane  $\Gamma_1$  that passes through the vertices  $v_{11} \in D_{11}$  and  $v_{12} \in D_{12}$ . Furthermore,  $a_{14}$  still lies on the rotated stabbing plane, as well as on  $\Gamma_2$ . In principle, it is possible to let  $\Gamma_1$  rotate further around the axis  $v_{11} \vee v_{12}$  until it hits a third vertex in either  $D_{13}$  or  $D_{14}$ , which would then completely determine the position of  $\Gamma_1$ . However, there is a risk that the resulting plane would cross  $D_{14}$  in a part that can no longer be reached by  $\Gamma_2$ . We will therefore proceed with  $\Gamma_2$ . But also when rotating  $\Gamma_2$  we run the risk that it will cross  $D_{14}$  in a part that is no longer reachable by  $\Gamma_1$ . To avoid this, we introduce the visual hull of  $D_{13}$  as seen from the axis  $v_{11} \vee v_{12}$ , and we replace  $D_{14}$  by

$$\hat{D}_{14} = D_{14} \cap H(v_{11} \vee v_{12}, D_{13}).$$

Clearly, for any point  $a_{14}$  in  $\hat{D}_{14}$ , we can always find a plane that passes through the axis  $v_{11} \vee v_{12}$  and that crosses  $D_{13}$ , since in (3) the visual hull was defined in a way to guarantee this. As mentioned previously, the visual hull can be constructed by joins and meets of domain vertices.

To proceed we choose arbitrary points  $a_{23} \in D_{23}$  and  $a_{24} \in D_{24}$  and let  $\Gamma_2$  rotate around the axis  $a_{23} \vee a_{24}$  until we hit a vertex in  $\hat{D}_{14}$  or  $D_{22}$ , for example,  $v_{22} \in D_{22}$ . Next, we let the plane rotate around the axis  $v_{22} \wedge a_{24}$  until we hit a second vertex, for example, the vertex  $v_{14}$  of  $\hat{D}_{14}$ . Note that this vertex may or may not be a vertex of the original domain  $D_{14}$ , but it is always constructible.

As soon as  $\Gamma_2$  passes through two vertices, we compute the visual hull of the third domain, chosen such that it is not equal to  $D_{24}$ , and as seen from the axis

passing through the two vertices already selected. For example, if the vertices are  $v_{22}$  and  $v_{14}$ , we compute

$$\hat{D}_{24} = D_{24} \cap H(v_{14} \vee v_{22}, D_{23})$$

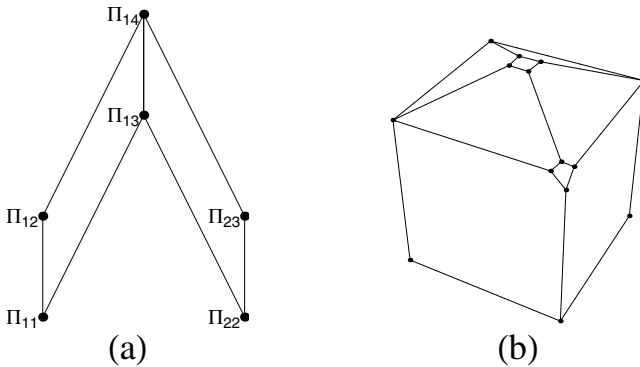
and replace  $D_{24}$  by  $\hat{D}_{24}$ . The previous step is now repeated until we have found  $m$  stabbing planes that contain two vertices each.

**Second Stage.** We note that for the last stabbing plane  $\Gamma_m$ , it is no longer a problem to rotate it around its two vertices, since the last domain in the configuration,  $D_{m4}$ , is not used in any  $(m + 1)$ -th concurrency relation. Hence, by rotating the last plane around the axis passing through the two vertices that have already been determined, we can determine a third vertex for the last plane. As soon as this vertex is known, we can go through the configuration in the opposite direction and determine a third vertex for each of the stabbing planes.

While doing this we have to replace the domains  $D_{i4}$  by their intersection with a stabbing plane. Suppose, for example, that there are three relations, as in Fig. 4, and that we have found vertices  $v_{32}, v_{33}, v_{34}$  that completely determine the third stabbing plane  $\Gamma_3 = v_{32} \vee v_{33} \vee v_{34}$ . To ensure that the second stabbing plane  $\Gamma_2$  will be able to meet  $\Gamma_3$  the third stabbing plane within the domain  $\hat{D}_{24}$ , we replace  $\hat{D}_{24}$  by

$$\tilde{D}_{24} = \hat{D}_{24} \cap (v_{32} \vee v_{33} \vee v_{34}).$$

Note that  $\tilde{D}_{24}$  is a planar polygon. Next we rotate the second last plane until it hits a third vertex, and replace the polytopes  $\hat{D}_{i4}$  when necessary by polygons



**Fig. 5.** Theorem 1 is not valid for the chain shown in (a) where 2 relations share more than one plane. (b) shows how this chain can arise in primal space. A polytope has two small facets, each adjacent to 4 larger facets. By changing the position of the large facets slightly, the small facets could disappear and be replaced by two vertices. However, two of the large facets are adjacent to both smaller facets. Each of the six large facets in (b) corresponds to a plane  $\Pi_{ij}$  in diagram (a). The planes  $\Pi_{13}$  and  $\Pi_{14}$  correspond to the large facets incident to both small facets.

$\tilde{D}_{i4}$ , until all the stabbing planes pass through constructible vertices in the dual space. Note that these vertices may either be vertices of the original domains, vertices of intersections of domains and visual hulls, or vertices of the polygons that result from taking the intersection of a domain, a visual hull and a stabbing plane already found.

After all the stabbing planes have been constructed, constructible primal planes can be pinned down as explained in Section 3.3.  $\square$

## 4 Conclusion

We have shown that if a solution of the concurrency problem exists, it can be constructed in a finite number of steps. We did not specify how many steps are necessary. Such more detailed analysis has been performed for line configurations in [10], but not yet for planes. Furthermore, the main result is only valid for connected chains where two concurrency relations share only one plane. In fact, when relations share two or more planes, as in Fig. 5, one can easily find examples where a solution cannot be constructed in a simple way from the support points. It remains to examine whether geometric constructions are possible for more complicated configurations than the ones studied here.

## References

1. He, P., Edalat, A.: Visual hull from imprecise polyhedral scene. In: Goesele, M., Matsushita, Y., Sagawa, R., Yang, R. (eds.) 3DIMPVT, pp. 164–171. IEEE (2011)
2. Laurentini, A.: The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.* 16(2), 150–162 (1994)
3. Petitjean, S.: A computational geometric approach to visual hulls. *Int. J. Comput. Geometry Appl.* 8(4), 407–436 (1998)
4. Labatut, P., Pons, J.P., Keriven, R.: Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In: *ICCV*, pp. 1–8. IEEE (2007)
5. Kim, H., Sakamoto, R., Kitahara, I., Toriyama, T., Kogure, K.: Robust Foreground Extraction Technique Using Gaussian Family Model and Multiple Thresholds. In: Yagi, Y., Kang, S.B., Kweon, I.S., Zha, H. (eds.) *ACCV 2007, Part I. LNCS*, vol. 4843, pp. 758–768. Springer, Heidelberg (2007)
6. Zivkovic, Z., van der Heijden, F.: Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters* 27(7), 773–780 (2006)
7. Kim, C.E.: Three-dimensional digital planes. *IEEE Trans. Pattern Anal. Machine Intell.* 6, 639–645 (1984)
8. Andres, E., Acharya, R., Sibata, C.: Discrete analytical hyperplanes. *Graphical Models and Image Processing* 59(5), 302–309 (1997)
9. Jamet, D., Toutant, J.L.: Minimal arithmetic thickness connecting discrete planes. *Discrete Applied Mathematics* 157(3), 500–509 (2009)
10. Veelaert, P., Heyvaert, M.: Reconstruction of Concurrent Lines from Leaning Points. In: Aggarwal, J.K., Barneva, R.P., Brimkov, V.E., Koroutchev, K.N., Korutcheva, E.R. (eds.) *IWCIA 2011. LNCS*, vol. 6636, pp. 182–193. Springer, Heidelberg (2011)

11. Coeurjolly, D., Sivignon, I., Dupont, F., Feschet, F., Chassery, J.M.: On digital plane preimage structure. *Discrete Applied Mathematics* 151(1-3), 78–92 (2005)
12. Veelaert, P.: Algorithms that measure parallelism and concurrency of lines in digital images. In: *Proc. of SPIE's Conference on Vision Geometry VIII*, Denver. SPIE, pp. 69–79 (1999)
13. Veelaert, P.: Concurrency of Line Segments in Uncertain Geometry. In: Braquelaire, A., Lachaud, J.-O., Vialard, A. (eds.) *DGCI 2002. LNCS*, vol. 2301, pp. 289–300. Springer, Heidelberg (2002)

# Scale Filtered Euclidean Medial Axis<sup>\*</sup>

Michał Postolski<sup>1,2</sup>, Michel Couprie<sup>1</sup>, and Marcin Janaszewski<sup>2</sup>

<sup>1</sup> Université Paris-Est, LIGM, Equipe A3SI, ESIEE,

Cité DESCARTES BP 99 93162 Noisy le Grand CEDEX, France

<sup>2</sup> Lodz University of Technology, Institute of Applied Computer Science

**Abstract.** We propose an Euclidean medial axis filtering method which generates subsets of Euclidean medial axis where filtering rate is controlled by one parameter. The method is inspired by Miklos', Giesen's and Pauly's scale axis method which preserves important features of an input object from shape understanding point of view even if they are at different scales. Our method overcomes the most important drawback of scale axis: scale axis is not, in general, a subset of Euclidean medial axis. It is even not necessarily a subset of the original shape. The method and its properties are presented in 2D space but it can be easily extended to any dimension. Experimental verification and comparison with a few previously introduced methods are also included.

**Keywords:** Filtered medial axis, discrete scale axis, shape representation, image analysis, stability.

## 1 Introduction

The notion of medial axis has been introduced by Blum in the 60s [4]. The medial axis of an object  $X$  is composed by the centers of the balls which are included in  $X$  but which are not fully included in any other ball included in  $X$ . This set of points is, by nature, centered in the object with respect to the distance which is used to define the notion of ball.

In the literature, different methods have been proposed to compute the medial axis approximately or exactly, for instance methods relying on discrete geometry [5,15,16,7], digital topology [13,25], mathematical morphology [22], computational geometry [3,20], partial differential equations [24], or level-sets [17]. In this work we focus on the discrete medial axis based on the Euclidean metric.

The medial axis is a very useful representation of the object and plays a major role in shape analysis in numerous applications, for example object recognition, registration or compression. From the medial axis points and associated ball radii, one can exactly reconstruct the original shape. However it can be hard or even impossible to use this tool effectively without first dealing with some problems, especially in discrete spaces and with noisy objects.

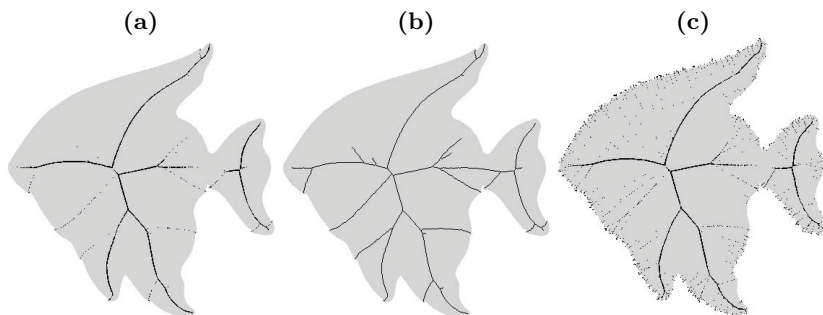
---

\* This work has been partially supported by the "ANR-2010-BLAN-0205 KIDICO" project and Polish National Science Centre grant No. N516 480640.



Firstly, the medial axis in discrete spaces has not, in general, the same topology as the original object. Solutions to this problem have been proposed by several authors, for instance [13,25,11]. They use discrete homotopic transformations guided and constrained by the medial axis, to obtain homotopic skeleton which contains the medial axis (see, Fig. 1). We do not consider these topological problems in the rest of the paper, and rely on this solution.

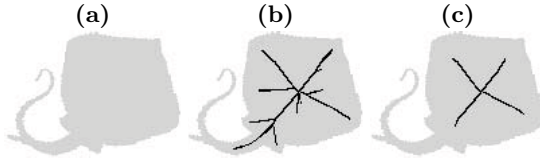
The second problem is sensitivity of the Euclidean medial axis to small contour perturbations (see, for example, Fig. 1). In other words, the medial axis is not stable under small perturbations of a shape: modifying a shape slightly (for example in terms of Hausdorff distance) can result in substantially different medial axes. This is a major difficulty when the medial axis is used in practical applications (e.g. shape recognition). A recent survey which summarises selected relevant studies dealing with this topic is presented in [2]. This fact, among others, explains why it is usually necessary to add a filtering step (or pruning step) to any method that aims at computing the medial axis and when a nonreversible but simplified description of binary objects is of interest.



**Fig. 1.** (a): a shape (in gray) and its Euclidean medial axis (in black); (b) the homotopic skeleton of the shape constrained by its Euclidean medial axis; (c) the same shape, but with small amount of noise added on the contour. The medial axis of the shape (c) is much more complicated than the medial axis of the shape (a).

The simplest strategy to filter the medial axis is to keep only points which are centers of maximal balls of at least a given diameter. Different criteria can be used to locally threshold and discard spurious medial axis points or branches: see [1,12], for methods based on the angle formed by the vectors to the closest points on the shape boundary, or the circumradius of these closest points [8,15].

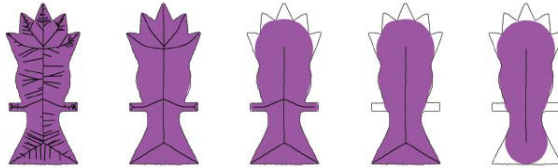
In these methods, a local information (that is, geometric information extracted from a single medial ball) is compared to a global parameter value to determine the importance of the corresponding medial axis point. However, it is well known that this local filtering can lead to remove small branches which might be important for the shape understanding (see Fig. 2) especially for shapes with features at different scales [2].



**Fig. 2.** (a): a shape  $X$  (in gray); (b): The filtered medial axis of  $X$  (in black) calculated by using algorithm [7]. The medial axis is not sufficiently filtered in the middle of the shape. However, we already start to lose the tail; (c) A more filtered medial axis of  $X$ . Now, the middle of the shape is well filtered. However, we lost all information about the tail.

A more complex criterion was proposed by [9]: they utilize information about ball importance in the shape with respect to all other balls by counting the number of object points inside a ball which are not covered by other balls. The medial axis point will be removed if the uncovered area of corresponding ball is too small.

In [19], the authors address this issue and propose an approach that put in relation local information and regional information, that is, the status of a ball is only influenced by the one of neighboring balls. Their method is based on the theory of the scale axis transform [14], and defines a whole family of medial representations at different levels of abstraction, called *scale axis representations* (see Fig. 3). For objects or scenes that include parts showing different scales, this method gives good results in many cases.



**Fig. 3.** Different scale axes of the same object (contoured), using different values of the scale parameter. In pink, the part of the object reconstructed from the filtered axis.

However, the scale axis representation is not free of drawbacks. The most important one is that the scale axis is not necessarily a subset of the Euclidean medial axis (see Fig. 4), it is even not necessarily a subset of the original shape.

In this paper we propose a new method for the Euclidean medial axis filtering (see section 4). Our proposition is inspired by the scale axis method (see section 3). However, as result we obtain a filtered Euclidean medial axis instead of a set of points that is not necessarily a subset of the latter. Furthermore, our method produces axes that preserve important features for shape understanding, even if they are at different scales. Therefore, our algorithm overcomes the most important drawbacks noticed in previously presented methods following a similar

approach. Moreover, the new method works in arbitrary dimensions. We evaluate experimentally its properties, and compare it with the previously introduced methods [9,12] (see section 5).

## 2 Basic Notions

In this section, we recall some basic geometrical and topological notions for binary images [6,18].

We denote by  $\mathbb{Z}$  the set of integers, by  $\mathbb{N}$  the set of nonnegative integers, and by  $\mathbb{N}_+$  the set of strictly positive integers. We denote by  $E$  the discrete space  $\mathbb{Z}^d$ . A point  $x$  in  $E$  is defined by  $(x_1, \dots, x_d)$  with  $x_i$  in  $\mathbb{Z}$ . Let  $x, y \in E$ , we denote by  $d(x, y)$  the Euclidean distance between  $x$  and  $y$ , that is,  $d(x, y) = ((x_1 - y_1)^2 + \dots + (x_d - y_d)^2)^{1/2}$ . In practice, the squared Euclidean distance is used in order to avoid floating numbers. Let  $Y \subset E$ , we denote by  $d(x, Y)$  the Euclidean distance between  $x$  and the set  $Y$ , that is,  $d(x, Y) = \min_{y \in Y} \{d(x, y)\}$ . Let  $X \subset E$  (the "object"), we denote by  $D_X$  the map from  $E$  to  $\mathbb{R}_+ \cup \{0\}$  which associates, to each point  $x$  of  $E$ , the value  $D_X(x) = d(x, \bar{X})$ , where  $\bar{X}$  denotes the complementary of  $X$  (the "background"). The map  $D_X$  is called the (*Euclidean distance map of X*). Let  $x \in E, r \in \mathbb{R}_+$ , we denote by  $B_r(x)$  the *ball of radius r centered on x*, defined by  $B_r(x) = \{y \in E, d(x, y) < r\}$ . Notice that, for any point  $x$  in  $X$ , the value  $D_X(x)$  is precisely the radius of a ball centered on  $x$  and included in  $X$ , which is not included in any other ball centered on  $x$  and included in  $X$ .

Now, let us recall the notion of medial axis (see also [21,25]). Let  $X \subseteq E$ . A ball  $B_r(x) \subseteq X$ , with  $x \in X$  and  $r \in \mathbb{N}_+$ , is *maximal for X* if it is not strictly included in any other ball included in  $X$ . The *medial axis of X*, denoted by  $MA(X)$ , is the set of the all couples  $(x, r)$  such that  $B_r(x)$  is a maximal ball for  $X$ .

Let  $X \subset E, Y \subset X$ , we denote by  $REDT_X(Y)$  the *reverse Euclidean distance transform* [9], defined by

$$REDT_X(Y) = \bigcup_{y \in Y} B_{D_X(y)}(y).$$

For exact and unfiltered  $MA(X)$  we have  $X = REDT_X(MA(X))$ .

## 3 Discrete Scale Axis

In this section, we adapt the notion of scale axis (see [19,14]), originally introduced in the continuous space and implemented in a framework of unions of balls, to the case of discrete grids. We denote by  $\mathbb{R}_+$  the set of strictly positive reals. Let  $X \subseteq E, x \in X, r \in \mathbb{N}_+$  and  $s \in \mathbb{R}_+$ . The parameter  $s$  is called the *scale factor*. We denote by  $X_s$  the *multiplicatively s-scaled shape*, defined by  $X_s = \bigcup_{(x,r) \in MA(X)} B_{rs}(x)$ . For  $s \geq 1$ , we denote by  $SAT_s(X)$  the *s-scale axis transform* of  $X$ , defined by

$$SAT_s(X) = \{(x, r/s) \mid (x, r) \in MA(X_s)\}.$$

The original algorithm to compute discrete scale axis, given by [19] in the framework of union of balls (UoBs), can be straightforwardly adapted to the case of  $\mathbb{Z}^d$  as follows. First, calculate the Euclidean medial axis of  $X$ . To do so, we use an efficient algorithm presented in [9]. Then multiply radius of each medial ball by the chosen scaling factor  $s$ .

In consequence small medial balls are covered completely by larger nearby balls since they are not important. On the other hand, small balls without larger balls in their neighborhood are not covered and will be preserved.

Next step is to reconstruct object  $X_s$  based on scaled radius values. Reconstruction can be made efficiently by reverse Euclidean distance transform (see section 2). Computing the medial axis of  $X_s$  achieves the simplification and  $MA(X_s)$  will be free of all covered balls, since these do not touch the boundary anymore and are thus no longer maximal. For  $s = 1$ , the scale axis is identical to unfiltered Euclidean medial axis. With increasing  $s$ , the scale axis gradually ignores less important features of  $X$  leading to successive simplifications of  $X_s$  and the scale axis structure.

The final step of the algorithm consists of rescaling the medial balls of  $MA(X_s)$  by a factor  $1/s$  to obtain the scale axis of  $X$ . Finally, discrete scale axis algorithm can be presented in the following pseudocode:

---

**Algorithm 1.** DiscreteScaleAxis(**Input**  $X, s$  **Output**  $SAT_s(X)$ )

---

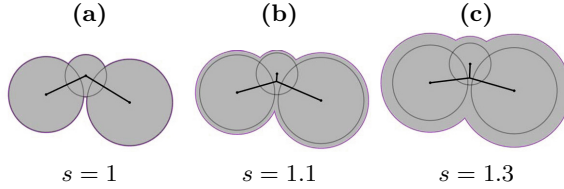
01. Compute  $MA(X)$
  02. Reconstruct  $X_s$
  03. Compute  $MA(X_s)$
  04. Compute  $SAT_s(X)$
- 

All four steps of DiscreteScaleAxis algorithm can be calculated in linear time in relation to  $\#X$ ,  $\#X_s$ ,  $\#X_s$  and  $\#MA(X_s)$  respectively, where  $\#X$  stands for cardinality of  $X$ . Therefore, computational complexity of the algorithm is  $O(\#X_s)$ .

## 4 The Scale Filtered Medial Axis

The crucial part of the method presented in the previous section, which is a source of problems ( $MA(X_s) \not\subseteq MA(X)$ ), is the reconstruction part after medial balls scaling and the need for generating a new medial axis from the scaled object (see Fig. 4). On the other hand, at first sight, this is the most important part of the algorithm since the medial axis simplification occurs in this part.

To filter  $MA(X)$  by removing centers of unimportant medial balls one must avoid reconstruction part and hold simplification property at the same time. Therefore, to solve this problem we assume that to make efficient filtration we



**Fig. 4.** (a): a shape  $X$  (in gray) and its Euclidean medial axis  $MA(X)$  (in black); (b): the multiplicatively 1.1-scaled shape of  $X$  and its 1.1-scaled axis; (c): the multiplicatively 1.3-scaled shape of  $X$  and its 1.3-scaled axis. In (b, c) we can see that scale axes are not subsets of  $MA(X)$ . In both cases, an additional branch even appears.

just need to decide which  $MA(X)$  points are not important and should be removed. Therefore, we do not generate a new object  $X_s$  and its  $MA(X_s)$ . In this way we obtain filtered Euclidean medial axis of  $X$  which is a subset of  $MA(X)$ .

This informal discussion motivates the following definition of the Scale Filtered Euclidean Medial Axis (SFEMA).

Let  $x \in X, r \in \mathbb{N}_+$ . We denote by  $B_r^X(x)$  the intersection of  $B_r(x)$  with  $X$ , that is,  $B_r^X(x) = \{y \in X \mid d(x, y) < r\}$ .

**Definition 1.** Let  $X \subseteq E$ , and  $s \in \mathbb{R}, s \geq 1$ . We denote by  $SFEMA_s(X)$  the Scale Filtered Euclidean Medial Axis of  $X$  defined by

$$SFEMA_s(X) = \{(x, r) \in MA(X) \mid B_{r/s}^X(x) \not\subseteq \bigcup_{(y,t) \in MA(X), t > r} B_{t/s}^X(y)\}.$$

Below, we give an algorithm to compute  $SFEMA_s(X)$  of a given object  $X \subseteq E$ .

The algorithm in line 02 performs sorting of medial axis elements, linearly in time using a counting sort [10]. In the following lines the algorithm performs two loops. The first one starts in line 04 and does  $\#X$  iterations. The next, nested loop, starts in line 06 and in worst case performs  $\#MA(X)$  iterations. Summarizing, computational complexity of SFEMA is  $O(\#X \#MA(X))$ .

Examples of  $SFEMA_s(X)$  for different scale factors  $s_i$ , are shown in Fig. 5.

Let us analyze properties and the major differences between the Miklos’s [19]  $s$ -scale axis and our  $s$ -scale filtered Euclidean medial axis. The most important property is that  $SFEMA_s(X)$  consists of  $MA(X)$  points only, that is, for all  $s \geq 1$ :  $SFEMA_s(X) \subseteq MA(X)$ . This property (inclusion property, for short) is essential in many applications of the medial axis. In Fig.4 we have shown an example of the Miklos’s scale axis where an additional branch even appears after filtering. Fig.6 shows another problem. The scale axis is too much simplified, loses important features of the object and is not included in the object. However,  $s$ -scale filtered medial axis holds inclusion property and permits to reconstruct the most of the original object.

The second interesting property relies on the notion of  $s$ -scale ball. If we want to simplify the object, using Miklos’s scale axis, for example, by removing a

**Algorithm 2.** SFEMA(Input  $X, s$  Output  $H$ )

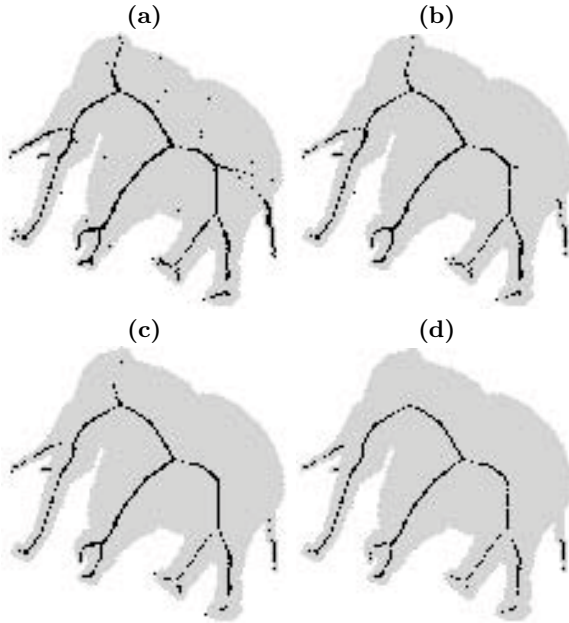
---

```

01.  $H \leftarrow \emptyset$ 
02.  $MA(X) \leftarrow \mathbf{EuclideanMedialAxis}(X)$ 
03. Let  $(x_1, r_1), \dots, (x_n, r_n)$  denote the elements of  $MA(X)$ 
    sorted in decreasing order of radii, that is,  $r_1 \geq \dots \geq r_n$ 
04. foreach  $p \in X$  do
05.      $i \leftarrow 1$ 
06.     while  $i \leq n$  and  $d(x_i, p) > sr_i$  do  $i \leftarrow i + 1$  end
07.     If  $i \leq n$  then
08.          $H \leftarrow H \cup \{(x_i, r_i)\}$ 
09.     end
10. end
11. return  $H$ 

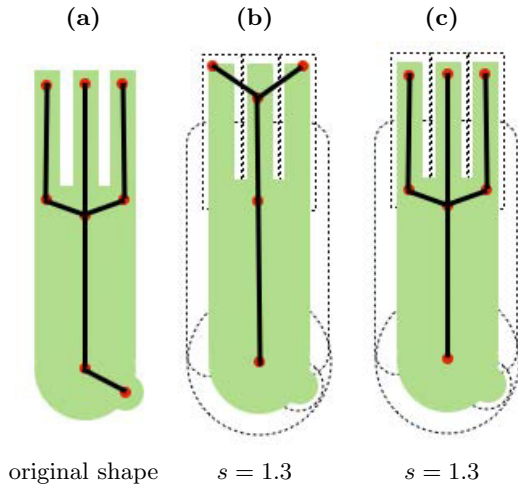
```

---

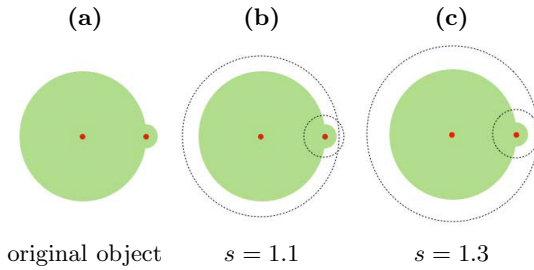


**Fig. 5.** (a): a shape  $X$  (in gray) and its Euclidean medial axis (in black); (b, c, d): the same shape and its  $SFEMA_{1.1}(X)$ ,  $SFEMA_{1.4}(X)$ ,  $SFEMA_{1.6}(X)$ , respectively. In all cases the elephant's tail, trunk, tusks and legs were considered as important and were not removed.

medial ball  $B_r(x)$ ,  $x \in X$ , the scale factor should be big enough that ball  $B_{r_s}(x)$  is included in one of other medial balls, that is,  $B_{r_s}(x) \subset B_{r_s}(y)$ ,  $y \in X$  (see Fig.7c), or in a union of such balls. In our algorithm, since we use notion of  $s$ -scaled ball, we only test inclusion inside  $X$  (see Fig.7b). This allows us to use



**Fig. 6.** (a): a set  $X$  (in green) and its  $MA(X)$  (in black); (b): the 1.3-scale axis of  $X$ ; (c): the 1.3-scale filtered medial axis of  $X$



**Fig. 7.** (a): a set  $X$  (in green) and its  $MA(X)$  (red dots); (b): multiplicatively scaled medial balls. The smaller ball is not fully covered by the bigger one after scaling. In scale axis representation both balls will be preserved. However, the bigger ball includes the smaller one inside set  $X$ . Therefore, the smaller ball will not exist in  $SFEMA_s(X)$ ; (c): multiplicatively scaled medial balls. The smaller ball is included in the bigger one. Therefore, it is neither in the scale axis nor in  $SFEMA_s(X)$ .

smaller scale factor. Therefore, we have better ability to control resulting  $s$ -scale filtered Euclidean medial axis.

## 5 Experiment Methodology and Results

In this section, we compare qualitatively and quantitatively properties of three medial axis filtering algorithms: discrete  $\lambda$ -medial axis (DLMA) [7], Euclidean medial axis filtered with the use of bisector function (BisEMA) [12] and SFEMA. In our experiments we use shapes from Kimia’s database [23].

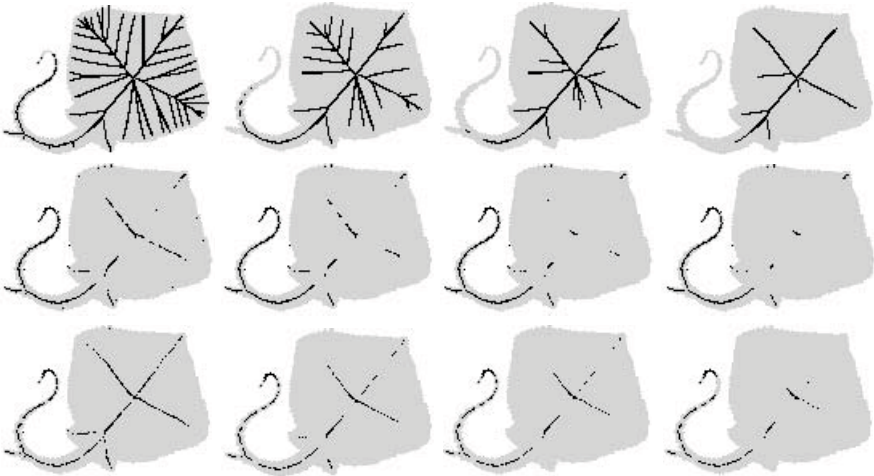
We first introduce some notions which help us to compare the algorithms. As it was stated in sec. 2, considering the exact and unfiltered  $MA(X)$ , the reverse Euclidean distance transform of  $MA(X)$  is equal to  $X$ . However, this property is no longer true if we consider filtered medial axes e.g. DLMA, BisEMA, SFEMA. Therefore, it is interesting to measure how much information about the original object is lost when we perform filtering. Considering a subset  $Y$  of  $MA(X)$ , we define:

$$R_X(Y) = \frac{|X \setminus REDT_X(Y)|}{|X|}.$$

We call  $R_X(Y)$  the (*normalised*) *residuals of Y*. Residuals give us a numerical evaluation of *reconstruction error*. Now we can set  $Y$  to different filtered medial axes, e.g. by using different methods or filtering parameters, and then evaluate which filtration is better in respect of ability to reconstruct the original object. The result  $R_X(Y)$  is a real value between 0 (perfect reconstruction) and 1 (bad reconstruction).

The normalised residuals factor is not enough to assess the quality of a filtered medial axis. It is difficult to compare different algorithms because filtering parameters of the algorithms have different meanings, therefore we introduce the *normalised medial axis size NS*.

Let denote by  $NS_X(Y)$  *normalised medial axis size* defined as a ratio of the number of the medial axis points to the number of object points:  $NS_X(Y) = \#Y/\#X$ .



**Fig. 8.** Medial axes in black superimpose to input object in grey color. Consecutive rows (from left to the right) contain results for DLMA, BisEMA and SFEMA respectively. Columns contain results for different values of normalised residuals: 0.01, 0.03, 0.05, 0.1 respectively.

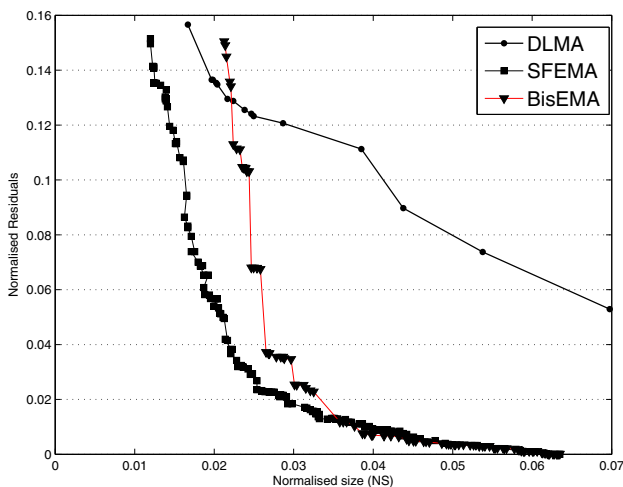


Now we can compare normalised residuals obtained using different methods for the same  $NS$ . In other words, we replace the parameters of the different medial axis filtering algorithms by only one parameter ( $NS$ ), which ensures a fair comparison.

Figure 8 presents DLMA, BisMA and SFEMA of an exemplary shape, extracted for several values of normalised residuals. The figure shows that SFEMA of smaller size than DLMA results in the same value of  $R$ . Moreover, in contrast to DLMA, SFEMA represents the most important fragments of an input object in different scales (see the tail in the last column of Fig. 8). SFEMA algorithm filters better than BisEMA some unimportant points close to the border of the sea devil main body.

**Table 1.** Average normalised residuals calculated for 18 representative shapes from Kimia’s database [23]. Lowest values are highlighted in gray.

NS	2D		
	2%	3%	5%
DLMA	0.2566	0.1717	0.0996
BisEMA	0.3268	0.1183	0.0775
SFEMA	0.0991	0.0562	0.0112



**Fig. 9.** Residuals as a function of normalised medial axis size for DLMA (curve with circles), BisEMA (curve with triangles) and SFEMA (curve with squares). Results generated for sea devil image (see Fig 8). Each marker (triangle, circle and square) represents parameters of one filtered medial axis. Lines has been added only to emphasise the trend of measurements.

The next important property of SFEMA can be concluded from Fig. 9 which shows accuracy of the object representation by filtered medial axis for different  $NSs$ . One can see that SFEMA obtains the smallest residuals value for most  $NSs$ . In other words SFEMA results with filtered medial axes which the most accurately represent the input object in different scales. Moreover Fig. 9 shows that SFEMA algorithm generates many more different filtered medial axes than DLMA algorithm does. This property is important when we are interested in multiscale representation of an input object. In this case, the number of different filtered medial axes generated with DLMA algorithm might be not enough. The above conclusions confirm results presented in Table 1. SFEMA has obtained the lowest mean normalised residuals for all  $NSs$ .

## 6 Conclusions

The article presents a new method for Euclidean medial axis filtering which possesses the following properties:

- generates subsets of Euclidean medial axis,
- filtering is based on only one parameter,
- generates filtered medial axes which preserve important parts of an input object in different scales,
- obtains smaller normalised residuals than other compared medial axis filtering algorithms,
- computation complexity of the algorithm is  $O(\#X\#MA(X))$ .

Future works will include the design of a more formalised framework for quantitative comparison of filtered medial axes at different scales. Using this framework the authors plan to perform more tests for 2D and 3D objects.

## References

1. Attali, D., Sanniti di Baja, G., Thiel, E.: Pruning Discrete and Semicontinuous Skeletons. In: Braccini, C., Vernazza, G., DeFloriani, L. (eds.) ICIAP 1995. LNCS, vol. 974, pp. 488–493. Springer, Heidelberg (1995)
2. Attali, D., Boissonnat, J.D., Edelsbrunner, H.: Stability and computation of the medial axis — a state-of-the-art report. In: Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration, pp. 109–125. Springer (2009)
3. Attali, D., Lachaud, J.: Delaunay conforming iso-surface, skeleton extraction and noise removal. Computational Geometry: Theory and Applications 19, 175–189 (2001)
4. Blum, H.: A transformation for extracting new descriptors of shape. In: Models for the Perception of Speech and Visual Form, pp. 362–380. MIT Press (1967)
5. Borgefors, G., Ragnemalm, I., Sanniti di Baja, G.: The euclidean distance transform: finding the local maxima and reconstructing the shape. In: Procs. of the 7th Scand. Conf. on Image Analysis, vol. 2, pp. 974–981 (1991)

6. Chassery, J., Montanvert, A.: Géométrie discrète. Hermès (1991)
7. Chaussard, J., Couprie, M., Talbot, H.: Robust skeletonization using the discrete lambda-medial axis. *Pattern Recognition Letters* 32(9), 1384–1394 (2011)
8. Chazal, F., Lieutier, A.: The  $\lambda$ -medial axis. *Graphical Models* 67(4), 304–331 (2005)
9. Coeurjolly, D., Montanvert, A.: Optimal separable algorithms to compute the reverse euclidean distance transformation and discrete medial axis in arbitrary dimension. *IEEE Trans. Pattern Anal. Mach. Intell.* 29(3), 437–448 (2007)
10. Cormen, T.H., Leiserson, C., Rivest, R.: *Introduction to algorithms*. MIT Press (1990)
11. Couprie, M., Bertrand, G.: Discrete topological transformations for image processing. In: Brimkov, V.E., Barneva, R.P. (eds.) *Digital Geometry Algorithms. Lecture Notes in Computational Vision and Biomechanics*, vol. 2, ch. 3, pp. 73–107. Springer (2012)
12. Couprie, M., Coeurjolly, D., Zrour, R.: Discrete bisector function and euclidean skeleton in 2d and 3d. *Image Vision Comput.* 25(10), 1543–1556 (2007)
13. Davies, E.R., Plummer, A.P.N.: Thinning algorithms: A critique and a new methodology. *Pattern Recognition* 14(1-6), 53–63 (1981)
14. Giesen, J., Miklos, B., Pauly, M., Wormser, C.: The scale axis transform. In: *Proceedings of the 25th Annual Symposium on Computational Geometry, SCG 2009*, pp. 106–115. ACM, New York (2009)
15. Hesselink, W., Roerdink, J.: Euclidean skeletons of digital image and volume data in linear time by the integer medial axis transform. *IEEE Trans. on PAMI* 30(12), 2204–2217 (2008)
16. Hulin, J.: *Axe Médian Discret: Propriétés Arithmétiques et Algorithmes*. Ph.D. thesis, Université Aix-Marseille II, Marseille (2009)
17. Kimmel, R., Shaked, D., Kiryati, N., Bruckstein, A.M.: Skeletonization via distance maps and level sets. *Computer Vision and Image Understanding* 62, 382–391 (1995)
18. Kong, T.Y., Rosenfeld, A.: Digital topology: Introduction and survey. *Computer Vision, Graphics, and Image Processing* 48(3), 357–393 (1989)
19. Miklos, B., Giesen, J., Pauly, M.: Discrete scale axis representations for 3d geometry. *ACM Trans. Graph.* 29, 101:1–101:10 (2010)
20. Ogniewicz, R., Kübler, O.: Hierarchic voronoi skeletons. *Pattern Recognition* 28(33), 343–359 (1995)
21. Rosenfeld, A.: *Digital image processing*. Academic Press (1982)
22. Serra, J.: *Image analysis and mathematical morphology*. Academic Press (1982)
23. Sharvit, D., Chan, J., Tek, H., Kimia, B.: Symmetry-based indexing of image databases. *Journal of Visual Communication and Image Representation* 9(4), 366–380 (1998)
24. Siddiqi, K., Bouix, S., Tannenbaum, A., Zucker, S.: The hamilton-jacobi skeleton. In: *International Conference on Computer Vision (ICCV)*, pp. 828–834 (1999)
25. Talbot, H., Vincent, L.: Euclidean skeletons and conditional bisectors. In: *Proceedings of VCIP 1992*. SPIE, vol. 1818, pp. 862–876 (1992)

# A Method for Feature Detection in Binary Tomography

Wagner Fortes<sup>1,2</sup> and K. Joost Batenburg<sup>1,3</sup>

<sup>1</sup> Centrum Wiskunde & Informatica, Amsterdam, The Netherlands

<sup>2</sup> Mathematical Institute, Leiden University, The Netherlands

<sup>3</sup> Vision Lab, University of Antwerp, Belgium

{wagner.fortes, joost.batenburg}@cwi.nl

**Abstract.** Binary tomography deals with the problem of reconstructing a binary image from its projections. Depending on properties of the unknown original image, the constraint that the image is binary enables accurate reconstructions from a relatively small number of projection angles. Even in cases when insufficient information is available to compute an accurate reconstruction of the complete image, it may still be possible to determine certain *features* of it, such as straight boundaries, or homogeneous regions. In this paper, we present a computational technique for discovering the possible presence of such features in the unknown original image. We present numerical experiments, showing that it is often possible to accurately identify the presence of certain features, even without a full reconstruction.

## 1 Introduction

Binary tomography deals with the problem of reconstructing a binary image from its projections. While accurate image reconstruction requires availability of a large number of projections for general grey scale images, knowledge about the fact the the unknown original image is binary can drastically reduce the number of projection angles needed for a detailed reconstruction in some cases.

A range of algorithms have been proposed for binary tomography [1–4]. Although each of these methods has demonstrated the ability to compute accurate reconstructions from a small number of projections in certain cases, none of these methods offer a *guarantee* that the reconstructed image is identical, or even similar to the unknown original image. In fact, one can state that giving such a guarantee will be impossible in general, as the reconstruction problem in binary tomography is known to be inherently *unstable*: a small change in the projection data can lead to a dramatic change in the (unique) reconstruction [5–7]. Moreover, several constructions are known for so-called *switching components*: binary images in which a selected set of zeros and ones can be interchanged, leading to a different image having the same projections [8, 9].

Even in cases when insufficient information is available to compute an accurate reconstruction of the complete image, it may still be possible to answer certain *questions* about the original image, or to determine certain *features* of it.

In [10], it was shown that connectivity and convexity properties can be derived – to some extent – directly from the projection data. It can also be desirable to know whether a certain boundary or homogeneous region can possibly exist in the unknown image, or not.

Even though finding a binary solution of the reconstruction problem is typically hard, it is often easier to prove that a solution *cannot* exist. For example, if the projections do not satisfy certain consistency conditions, a solution will certainly not exist. General consistency conditions for the Radon transform are presented in [11], while a detailed analysis of consistency conditions for the grid model in discrete tomography can be found in [12]. A particular condition for the existence of binary solutions is given in [13], which will be used and extended throughout the present paper.

In this article, we extend the general idea of consistency to the detection whether or not certain *substructures* can exist in the original image. We present a computational technique for discovering the possible presence of certain features (e.g., blobs, edges). For each feature, a *probe* structure is defined, which can detect that particular feature. Based on an analysis of the existence of binary solutions of the reconstruction problem, our technique can prove, in certain cases, if the probed feature *cannot* exist in a given region of the original image. Our approach is independent of a particular reconstructed image or reconstruction method.

This paper is structured as follows. In Section 2, the basic model and notation are introduced. In Section 3, the basic idea of a *probe image* is presented and formally defined. Section 4 covers various algorithms that can be used to prove – in certain cases – that a given probe image cannot be present in the unknown original image. Section 5 presents a series of simulation experiments that was performed to obtain a first assessment of the capabilities of the proposed method. Conclusions are drawn in Section 6.

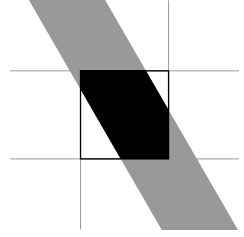
## 2 Basic Notation and Model

Throughout the discrete tomography literature, several imaging models have been considered: the grid model, the strip model, the line model, etc. [14, section 7.4.1]. In this paper we focus on the strip model, but our approach can be used for other projection models as well.

In the strip model, a projection is computed by considering a set of parallel strips in a given direction and for each strip computing the weighted sum of all the pixels which intersect that strip with a weight equal to the intersection area of the strip and the pixel (see Fig. 1).

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*, which correspond to unit squares in the strip 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 strip model its entries are real values in  $[0, 1]$ . 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 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.



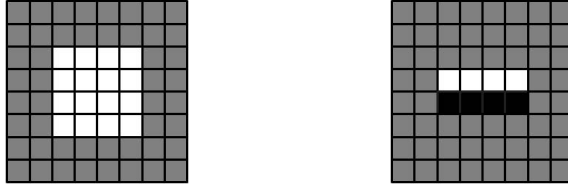
**Fig. 1.** The strip model

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.

### 3 Probe Structure

We now introduce the concept of a *probe image*. A probe image is represented by a vector  $\mathbf{v} = (v_i) \in \{0, 1, ?\}^n$ . We say that a binary image  $\bar{\mathbf{x}}$  *satisfies* the probe image  $\mathbf{v}$  iff  $\bar{x}_i = v_i$  whenever  $v_i \in \{0, 1\}$ . This relation is denoted by the predicate  $F(\mathbf{v}, \bar{\mathbf{x}})$ . In other words, the zeros and ones in the probe image prescribe the values of the corresponding pixels in  $\bar{\mathbf{x}}$ , while a pixel value of '?' in the probe image allow any pixel value in the corresponding pixel of  $\bar{\mathbf{x}}$ . We denote the set of all images satisfying a probe image  $\mathbf{v}$  by  $\mathcal{F}(\mathbf{v}) = \{\bar{\mathbf{x}} \in \{0, 1\}^n : F(\mathbf{v}, \bar{\mathbf{x}})\}$ . For any probe image  $\mathbf{v}$ , define  $s(\mathbf{v}) = \#\{1 \leq i \leq n : v_i \neq '?'\}$ , the total number of 0's and 1's in the probe image.

Suppose that we want to know if the unknown original image may contain a certain homogeneous region of 1's (i.e., white pixels). We then define a probe image  $\bar{\mathbf{v}}$  that has such a homogeneous region, and contains the '?' symbol in all pixels that are not in this region (see Fig. 2a). The question whether there exists a binary solution of the tomography problem that has such a region can then be rephrased as a check whether the set  $\bar{S}_{\mathbf{W}}(\mathbf{p}) \cap \mathcal{F}(\bar{\mathbf{v}})$  is empty or not. Similarly, one can define an *edge detection* probe image such as shown in Fig. 2b. Any image that has a horizontal edge at the indicated position, consisting of a black strip of background pixels and a white strip of foreground pixels (i.e., an edge at the bottom of a white region), will be part of the set  $\mathcal{F}(\bar{\mathbf{v}})$  for this probe image  $\bar{\mathbf{v}}$ . This brings us to the central problem considered in this paper:



**Fig. 2.** Two probe images. The pixels are colored as follows: black (0), grey (?), white (1). Left: homogeneous white region; right: horizontal edge at the bottom of a white region.

*Problem 1.* (Probe problem). Let  $\mathbf{W} \in \mathbb{R}^{m \times n}$  be a given projection matrix and let  $\mathbf{v} \in \{0, 1, ?\}^n$  be a given probe image. Determine if  $\bar{S}_{\mathbf{W}}(\mathbf{p}) \cap \mathcal{F}(\mathbf{v}) = \emptyset$ .

If the intersection between the solution set of the tomography problem and the set of images that satisfy the probe image is not empty, we cannot conclude if the unknown original image satisfies the probe image. However, if the intersection between both sets is empty, we can conclude that no binary solution exists that has the probed feature. As we will see in the next sections, one can often prove that the answer to Problem 1 is “yes”, even without enumerating the set  $\bar{S}_{\mathbf{W}}(\mathbf{p})$  of binary solutions of the reconstruction problem.

Now consider the system of equations

$$\begin{pmatrix} | & & | \\ \mathbf{w}_1 & \cdots & \mathbf{w}_n \\ | & & | \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \mathbf{p}, \tag{1}$$

where  $\mathbf{w}_i$  denotes the  $i$ th column vector of  $\mathbf{W}$ . We now define the operation of *fixing* a pixel  $x_i$  at value  $v_i \in \mathbb{R}$ . It transforms the system (1) into the new system

$$\begin{pmatrix} | & & | & & | \\ \mathbf{w}_1 & \cdots & \mathbf{w}_{i-1} & \mathbf{w}_{i+1} & \cdots & \mathbf{w}_n \\ | & & | & & | \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_{i-1} \\ x_{i+1} \\ \vdots \\ x_n \end{pmatrix} = \mathbf{p} - v_i \mathbf{w}_i. \tag{2}$$

The new system has the same number of equations as the original system, whereas the number of variables is decreased by one. The *fixing* operation can be performed for more than one pixel at time.

**Proposition 1.** Let  $\mathbf{W} \in \mathbb{R}^{m \times n}$  be a given projection matrix and let  $\mathbf{v} \in \{0, 1, ?\}^n$  be a given probe image. Let  $\mathbf{R}\mathbf{y} = \mathbf{q}$  be the linear system that is obtained by fixing all pixels  $x_i$  to value  $v_i$  whenever  $v_i \in \{0, 1\}$ . Then solving Problem 1 is equivalent to checking whether  $\bar{S}_{\mathbf{R}}(\mathbf{q}) = \emptyset$ .

We call the linear system formed in Prop. 1 the *reduced linear system* corresponding to the probe image  $\mathbf{v}$ .

### 4 Partially Solving the Probe Problem

As noted in the previous section, the probe problem can be rephrased as the question whether or not the reduced linear system has a binary solution. In [13], the authors present a sufficient condition for the existence of binary solutions of a given linear system  $\mathbf{W}\mathbf{x} = \mathbf{p}$  which satisfies  $\sum_{i=1}^m w_{ij} = k$  for all  $j = 1, \dots, n$ . In summary, it is proved that all binary solutions of this linear system lie on a hypersphere centered in the minimum norm solution  $\mathbf{x}^*$  and having radius  $\mathcal{R}(\mathbf{p}, \mathbf{x}^*) = \sqrt{\frac{\sum_{i=1}^m p_i}{k} - \|\mathbf{x}^*\|_2^2}$ . If the binary vector closest to  $\mathbf{x}^*$  is outside this hypersphere then the given linear system contains no binary solutions:

**Theorem 1.** *Let  $\mathbf{x}^* = \mathbf{W}^\dagger \mathbf{p}$ , where  $\mathbf{W}^\dagger$  denotes the Moore-Penrose inverse of  $\mathbf{W}$  [15]. For  $\alpha \in \mathbb{R}$ , let  $\rho(\alpha) = \min(|\alpha|, |1 - \alpha|)$  and put  $T(\mathbf{x}^*) = \sqrt{\sum_{i=1}^n \rho^2(x_i^*)}$ . If  $\mathcal{R}(\mathbf{p}, \mathbf{x}^*) < T(\mathbf{x}^*)$ , then  $\bar{S}_{\mathbf{W}}(\mathbf{p}) = \emptyset$ .*

*Proof.* See [13].

In the remainder of this section we present two related techniques for proving that the unknown original image *does not* satisfy a given probe image  $\mathbf{v}$ . Both methods use variants of Theorem 1.

#### 4.1 Probing by Analyzing the Binary Solutions of the Reduced Linear System

Let  $\mathbf{v}$  be a given probe image. We now analyze the reduced linear system  $\mathbf{R}\mathbf{y} = \mathbf{q}$  corresponding to  $\mathbf{v}$ , in terms of the existence of binary solutions, following the idea of Theorem 1.

Let  $\mathbf{y}^* = \mathbf{R}^\dagger \mathbf{q}$  and  $\rho(\alpha) = \min(|\alpha|, |1 - \alpha|)$ . Put  $T(\mathbf{y}^*) = \sqrt{\sum_{i=1}^n \rho^2(y_i^*)}$  and define  $\mathcal{T}(\mathbf{y}^*) = \{\bar{\mathbf{r}} \in \{0, 1\}^{n-s(\mathbf{v})} : \|\bar{\mathbf{r}} - \mathbf{y}^*\|_2 = T(\mathbf{y}^*)\}$ . Also, let  $\bar{\mathbf{r}} \in \mathcal{T}(\mathbf{y}^*)$ , i.e.,  $\bar{\mathbf{r}}$  is among the binary vectors that are nearest to  $\mathbf{y}^*$  in the Euclidean sense. Vector  $\bar{\mathbf{r}}$  can be easily computed by rounding the entries of  $\mathbf{y}^*$  to their nearest value in the set  $\{0, 1\}$ . Despite  $\bar{\mathbf{r}}$  may not be unique, any choice of  $\bar{\mathbf{r}}$  yields the same results in this context.

Rewriting Theorem 1 in the framework of identifying the existence of binary images satisfying a given probe image  $\mathbf{v}$ , we have:

**Theorem 2.** *Let  $\mathbf{y}^* = \mathbf{R}^\dagger \mathbf{q}$  and  $\bar{\mathbf{r}} \in \mathcal{T}(\mathbf{y}^*)$ . If  $\|\bar{\mathbf{r}} - \mathbf{y}^*\|_2 > \sqrt{\frac{\sum_{i=1}^m q_i}{k} - \|\mathbf{y}^*\|_2^2}$  then the original system  $\mathbf{W}\mathbf{x} = \mathbf{p}$  does not have a binary solution which satisfies  $\mathbf{v}$ .*

*Proof.* From Theorem 1, we know that if  $\|\bar{\mathbf{r}} - \mathbf{y}^*\|_2 > \sqrt{\frac{\sum_{i=1}^m q_i}{k} - \|\mathbf{y}^*\|_2^2}$  then there is no binary vector satisfying  $\mathbf{R}\mathbf{y} = \mathbf{p}$ . Hence, there is no  $\bar{\mathbf{x}} \in \bar{S}_{\mathbf{W}}(\mathbf{p})$  that satisfies  $\mathbf{v}$ .



### 4.2 Probing by Analyzing the Binary Solutions of the Original Linear System

Using an idea similar to what was used in the previous subsection, we now analyze the consistency of the original linear system with respect to binary solutions. However, instead of using  $\bar{\mathbf{r}}$ , the binary vector closest to the minimum norm solution  $\mathbf{x}^*$ , we define  $\tilde{\mathbf{r}}$  as the binary vector, satisfying the probe image  $\mathbf{v}$ , which is closest to  $\mathbf{x}^*$ .

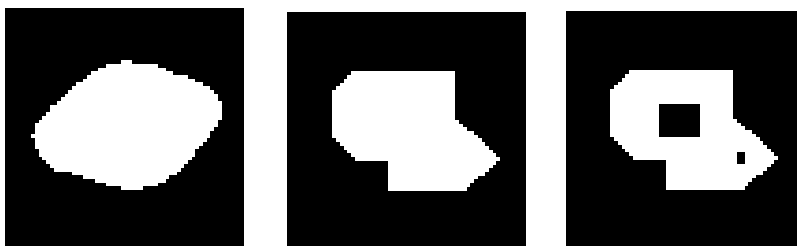
**Theorem 3.** *Let  $\mathbf{x}^* = \mathbf{W}^\dagger \mathbf{p}$  and  $\bar{\mathbf{r}} \in \mathcal{T}(\mathbf{x}^*)$ . For  $i = 1, \dots, n$ , define  $\tilde{r}_i = \bar{v}_i$  if  $v_i \neq 0$  and  $\tilde{r}_i = \bar{r}_i$  otherwise. If  $\|\tilde{\mathbf{r}} - \mathbf{x}^*\|_2 > \sqrt{\frac{\sum_{i=1}^m p_i}{k} - \|\mathbf{x}^*\|_2^2}$ , then  $\mathbf{v}$  is not satisfied by any  $\tilde{\mathbf{x}} \in \bar{S}_{\mathbf{W}}(\mathbf{p})$ .*

*Proof.* The vector  $\tilde{\mathbf{r}}$  is the binary image which contains the structure of the probe image that is closest to  $\mathbf{x}^*$ . If  $\tilde{\mathbf{r}}$  is out of the hypersphere containing all binary solutions of  $\mathbf{W}\mathbf{x} = \mathbf{p}$  (see [13]), then there is no binary image satisfying the probe image  $\mathbf{v}$  that is on this hypersphere. Therefore  $\mathbf{v}$  is not satisfied by any binary solution of  $\mathbf{W}\mathbf{x} = \mathbf{p}$ .

## 5 Numerical Experiments

Although the two techniques from Section 4 can detect sufficient conditions for the non-existence of binary solutions of the reconstruction problem that satisfy the given probe image, an empirical study is needed to determine the usefulness of the proposed methods for actual tomography data. A series of experiments was performed to investigate the presented method, for three different phantom images using a variable number of projections. The experiments are all based on simulated projection data obtained by computing the projections of the test images (so-called *phantoms*) in Fig. 3:

For the experiments, we have used probe images that only consider 0's and 1's inside a square sub-image of size  $8 \times 8$  pixels. This subregion is then moved across the full image region, scanning the possible presence of the probe structure throughout the image of size  $64 \times 64$  pixels.



(a) Phantom 1,  $64 \times 64$  (b) Phantom 2,  $64 \times 64$  (c) Phantom 3,  $64 \times 64$

**Fig. 3.** Original phantom images used for the experiments

For each experiment, both techniques from Section 4 were used, checking whether any of these two methods can prove that the probe structure cannot occur in a particular region of the unknown original image.

To compute the shortest least-squares solutions of the linear systems involved in the methods of Section 4, the CGLS algorithm was used. We refer to [13] for details.

In the following subsections, we consider experiments for two different probe structures, for detecting homogeneous regions and horizontal edges, respectively.

## 5.1 Homogeneous Regions

In this section, we focus on the identification of square homogeneous regions in the unknown original image. Two types of probe images were defined: a square  $8 \times 8$  region of 1's (white pixels) surrounded by '?' pixels, and a square  $8 \times 8$  region of black pixels, also surrounded by '?' pixels. These square regions were then moved across the full  $64 \times 64$  image region to determine at each location whether such a homogeneous black or white square can possibly occur in the binary solution set of the tomography problem.

For each probe image we are able to define a *status* based on the results obtained by applying the presented methods with the two different types of probe images. We define the status **forbidden** for a probe image which, according to the methods, have no binary solution satisfying it. We also define the status **allowed** for a probe image in which the methods could not determine whether there exists a binary solution satisfying this probe image.

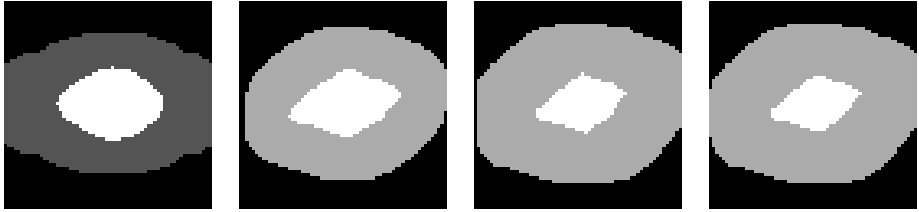
The results for a given phantom image leads to a new 2D greyscale image, which represents – for each position of the probe region – the outcome for both probe types, as follows:

- If the black region is **allowed** and the white region is **forbidden** then associate a *black* color;
- If the black region is **forbidden** and the white region is **allowed** then associate a *white* color;
- If the black region is **allowed** and the white region is **allowed** then associate a *light grey* color;
- If the black region is **forbidden** and the white region is **forbidden** then associate a *dark grey* color;

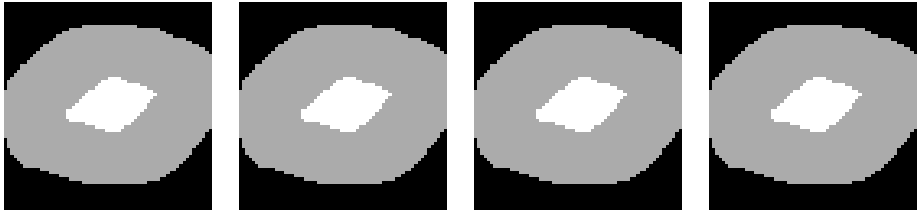
The resulting greyscale images are shown in Fig. 4 for Phantoms 1 and 2, depicting results for an increasing number of projection angles. It can be observed that as the number of angles grows, the results of the probe experiments provide an increasingly accurate view of the true presence of homogeneous regions in the phantom image.

## 5.2 Horizontal Edges

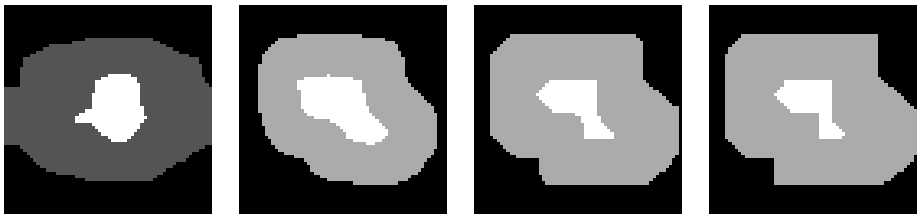
The goal of this section is to identify straight horizontal edges which could be present in the original image. We use the term horizontal edges to indicate



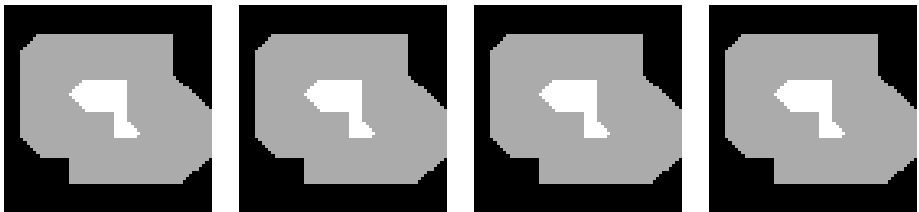
(a) Phantom 1. Number of projection angles: 2, 4, 8 and 12



(b) Phantom 1. Number of projection angles: 16, 20, 24, and 28



(c) Phantom 2. Number of projection angles: 2, 4, 8 and 12

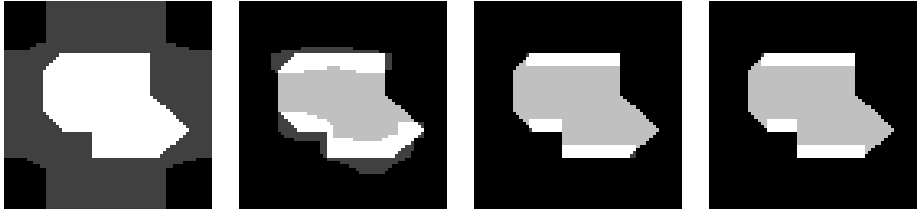


(d) Phantom 2. Number of projection angles: 16, 20, 24, and 28

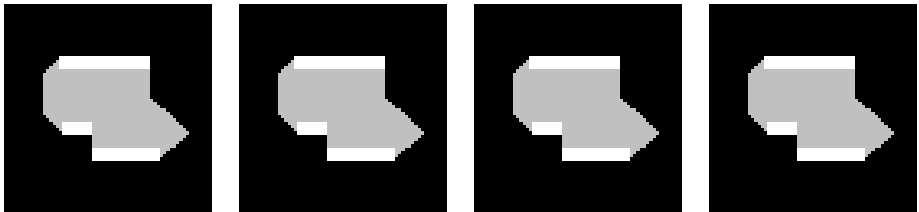
**Fig. 4.** Homogeneous region status for the phantom images of dimension  $64 \times 64$ 

horizontally adjacent pixels with intensity 1 (white) which are vertically adjacent to the same number of horizontally adjacent pixels with intensity 0 (black color). So, we define a square probe structure of size  $8 \times 8$  such that the pixels in the upper half of the square are set to 0 and the pixels in the lower half of the square are set to 1. The vertically mirrored version of this probe structure was also used to detect edges at the bottom of an object.

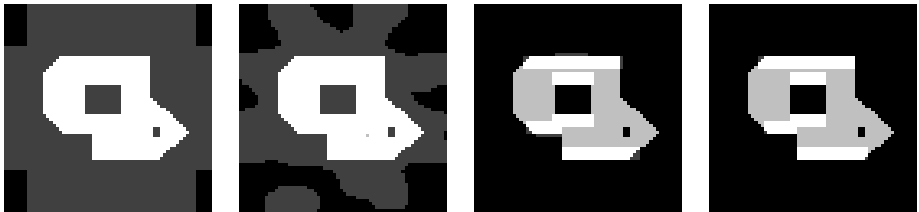
Similar to the previous section, the results for this probe structure give rise to a new greyscale image, defined as follows. Starting from a completely black image, if at a certain position for the probe structure no unsatisfiability is detected, the “white” part of the edge (corresponding to the interior of the object) is colored white in the output image if it is also white in the original image and dark grey if it is black in the original image. The “black” part of the edge (corresponding to the outside of the object) is colored black in the output image if it is also black in the original image



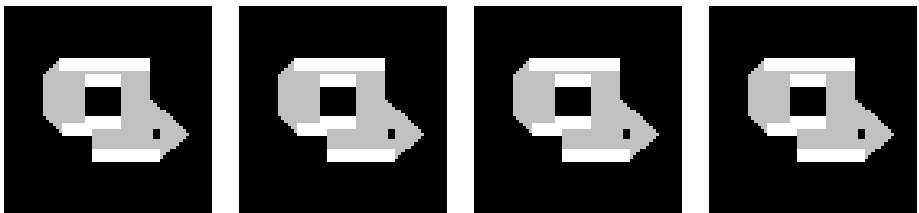
(a) Phantom 2. Number of projection angles: 2, 4, 8 and 12



(b) Phantom 2. Number of projection angles: 16, 20, 24, and 28



(c) Phantom 3. Number of projection angles: 2, 4, 8 and 12



(d) Phantom 3. Number of projection angles: 16, 20, 24, and 28

**Fig. 5.** Possible edges for the phantom images of dimension  $64 \times 64$

in the original image and light grey if it is white in the original image. Also, if at a certain position for the probe structure the unsatisfiability is detected but there are white pixels in this region in the original image, then those pixels are colored as light grey. The results of this procedure are shown in Fig. 5, which identifies the regions that could be edges according to our results, for a varying number of projection angles. Again, we see that as the number of projections increases, the results of the probe experiments provide an increasingly accurate view of the true presence of horizontal edges in the phantom image.

## 6 Outlook and Conclusion

In this article we have proposed a novel approach for obtaining information about an object from a small number of its projections. By using necessary conditions for the existence of binary solutions of the tomography problem, and combining these with probe images for particular substructures of the image, it can be determined whether such a substructure can possibly occur, or whether it can certainly not occur in the unknown original image.

The experimental results for a limited set of simulation experiments show that this approach can indeed lead to the recovery of substantial information about the original image, without resorting to a particular, possibly non-unique reconstruction.

More research in this direction will be necessary to determine what the limitations are of the proposed method, and how it compares to image analysis algorithms that try to find the structure directly in a reconstructed image.

**Acknowledgements.** W.F. acknowledges support from the Erasmus Mundus program of the European Union, and from the University of Leiden. K.J.B. was supported by the Netherlands Organisation for Scientific Research (NWO), programme 639.072.005.

## References

1. Batenburg, K.J.: A network flow algorithm for reconstructing binary images from continuous X-rays. *J. Math. Im. Vision* 30(3), 231–248 (2008)
2. Batenburg, K.J., Sijbers, J.: Dart: a practical reconstruction algorithm for discrete tomography. *IEEE Trans. Image Processing* 20(9), 2542–2553 (2011)
3. Herman, G.T.: *Fundamentals of Computerized Tomography: Image reconstruction from projections*. Springer (2009)
4. Schüle, T., Schnörr, C., Weber, S., Hornegger, J.: Discrete tomography by convex-concave regularization and D.C. programming. *Discr. Appl. Math.* 151, 229–243 (2005)
5. Alpers, A., Gritzmam, P.: On stability, error correction, and noise compensation in discrete tomography. *SIAM Journal on Discrete Mathematics* 20(1), 227–239 (2006)
6. Alpers, A., Brunetti, S.: Stability results for the reconstruction of binary pictures from two projections. *Image and Vision Computing* 25(10), 1599–1608 (2007)

7. Van Dalen, B.: Stability results for uniquely determined sets from two directions in discrete tomography. *Discrete Mathematics* 309, 3905–3916 (2009)
8. Hajdu, L., Tijdeman, R.: Algebraic aspects of discrete tomography. *J. Reine Angew. Math.* 534, 119–128 (2001)
9. Herman, G.T., Kuba, A. (eds.): *Discrete Tomography: Foundations, Algorithms and Applications*. Birkhäuser, Boston (1999)
10. Gara, M., Tasi, T.S., Balazs, P.: Learning connectedness and convexity of binary images from their projections. *Pure Math. Appl.* 20(1-2), 27–48 (2009)
11. Helgason, S.: *The Radon transform*. Birkhäuser, Boston (1980)
12. Stolk, A., Batenburg, K.J.: An algebraic framework for discrete tomography: Revealing the structure of dependencies. *SIAM J. Discrete Math.* 24(3), 1056–1079 (2010)
13. Batenburg, K.J., Fortes, W., Hajdu, L., Tijdeman, R.: Bounds on the Difference between Reconstructions in Binary Tomography. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) *DGCI 2011*. LNCS, vol. 6607, pp. 369–380. Springer, Heidelberg (2011)
14. Kak, A.C., Slaney, M.: *Principles of Computerized Tomographic Imaging*. SIAM (2001)
15. Björck, Å.: *Numerical methods for least square problems*. SIAM, Linköping University, Sweden (1996)

# Knot Segmentation in Noisy 3D Images of Wood

A. Krähenbühl, B. Kerautret, and I. Debled-Rennesson

Université de Lorraine, LORIA, Adagio team, UMR 7503, 54506, Nancy, France

**Abstract.** Resolving a 3D segmentation problem is a common challenge in the domain of digital medical imaging. In this work, we focus on another original application domain: the 3D images of wood stem. At first sight, the nature of wood image looks easier to segment than classical medical image. However, the presence in the wood of a wet area called sapwood remains an actual challenge to perform an efficient segmentation. This paper introduces a first general solution to perform knot segmentation on wood with sapwood. The main idea of this work is to exploit the simple geometric properties of wood through an original combination of discrete connected component extractions, 2D contour detection and dominant point detection. The final segmentation algorithm is very fast and allows to extract several geometrical knot features.

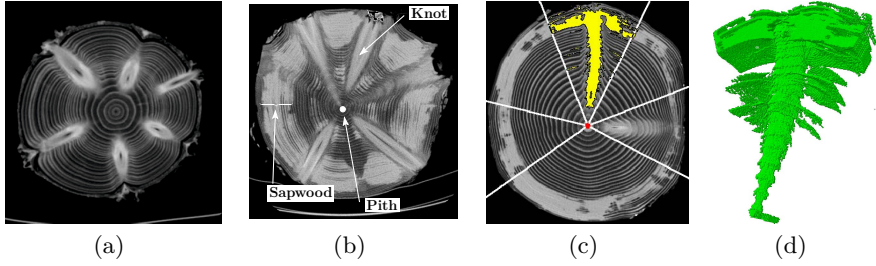
**Keywords:** 3D segmentation, dominant points, histogram, geometrical features, wood knot.

## 1 Introduction

Outside the classical medical applications, 3D digital imaging systems like X-Ray Computer Tomography are an interesting way for biologists to analyze wood. Even less frequent, these original images could offer the possibility to access to numerous geometric informations about wood knots [12]. A wood knot is the young part of a branch included in the wood stem (see Fig. 1(b)). Sawmills are also interested by knot segmentation in 3D images to optimize the cutting decisions of wood planks. They expect to improve at the same time the wood plank appearance and the productivity.

By nature, wood structures are simpler to segment than medical images. At first view, the extraction of wood knot does not seem difficult on ideal configurations as in Fig. 1(a). On the contrary, the image 1(b) remains an important problem to apply segmentation. A simple threshold is no more possible since sapwood and knot are connected and of similar intensity. The problem of sapwood is a well known major difficulty.

More precisely, several authors have tried to remove the limitations induced by sapwood. We can cite Andreu and Rinnhofer who propose a specific method based on knot model to segment knots [5]. Their approach is robust on sapwood but is limited to the Norway spruce. Moreover, it can only detect knots with particular orientation specific to the species. This approach can not detect the correct knot structure inside sapwood. Another approach was proposed by Aguilera *et al.* [2]. They use a 2D deformable model with simulated annealing. Their



**Fig. 1.** Illustration of different knot qualities in X-Ray images. Image 1(a) is an ideal configuration without sapwood, image 1(b) is a noisy version with sapwood. Images 1(c) and 1(d) show the limitations of the previous proposed approach [9].

approach can give results in presence of sapwood but an important separation is visible between knot and sapwood on the examples of their experiments. The proposed method also suffers from the setting of the deformable model parameters. Moreover, it is not automatic: the deformable model must be manually initialized. We can also refer to the work of Nordmark [14]. He proposes to use neural networks to solve the knot detection problem in presence of sapwood. This original solution is interesting but presents the main inconvenients to be very slow and does not always provide precise results. Note that other classical segmentation approaches, like the 3D deformable models, are also not able to perform knot segmentation in presence of sapwood (see for instance the segmentation comparison in [9] or the one of Fig 10(f)).

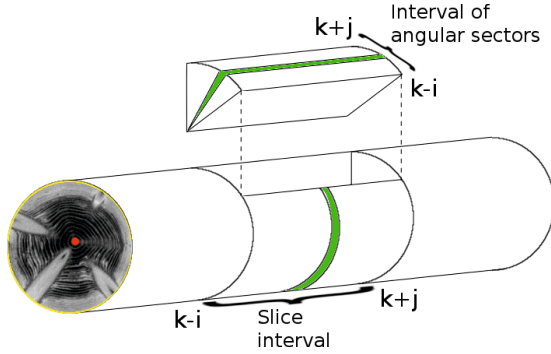
In previous work, we proposed a sapwood robust approach to detect position and orientation of knots [9]. This first step is only limited to the detection and does not segment the knots. In particular, the limits of such approach are illustrated on Fig. 1(c) and 1(d). The contribution of this new paper is to advance further than the previous detection by performing a real segmentation even in presence of sapwood. The main idea of the proposed method is to exploit geometric information analyzed from the discrete contour of 2D images. Through this method, we can integrate an *a priori* shape knowledge on knot and sapwood while remaining efficient.

Before introducing this new solution, we summarize in the following section the previous work of knot area detection which represents an important step in the proposed method. Afterwards, we detail our knot segmentation method. Finally, we present our results and comparisons with results of recent segmentation algorithms.

## 2 Knot Areas Detection

In previous work, we present an histogram based method to isolate each tree knot in an angular interval [9]. Since our work relies on this detection we recall here the main steps of the algorithm.





**Fig. 2.** Schema of the detection of slice intervals and angular intervals. In green a slice and an angular sector corresponding to a peak in the z-motion histograms and pie-charts. The  $[k - i, k + j]$  intervals are the computed intervals from each peaks.

We work with a 3D grey level image  $I$  of  $N \times M \times K$  resolution. A slice  $S_k$  of size  $N \times M$  is a subimage of  $I$  corresponding to a cross-section of the tree.  $S_1$  and  $S_K$  correspond respectively to the bottom and the top of tree when it is scanned from bottom to top. We work also with the pith. Biologically, the pith is the tree center, the growth rings center and the most important for us, the knot origin. To localize the pith, we use an algorithm proposed by Fleur Longuetaud [11] based on the growth rings detection. The pith position is defined as the intersection point of the lines perpendicular to the growth rings. We obtain one pixel by slice. The pith position will be used like center of all the angular sectors.

**The z-Motion.** By sliding the slices, we can observe that knots move from the pith to the bark. Only the knots produce big motions due to a big contrast with softwood. We name this motion the z-motion.

**Definition 1.** Let  $(S_k)_{k \in [1, K]}$  be a set of  $K$  slices. The z-motion slice  $Z_k$  is defined as the absolute value of intensity variation between the two consecutive slices  $S_{k-1}$  and  $S_k$  :

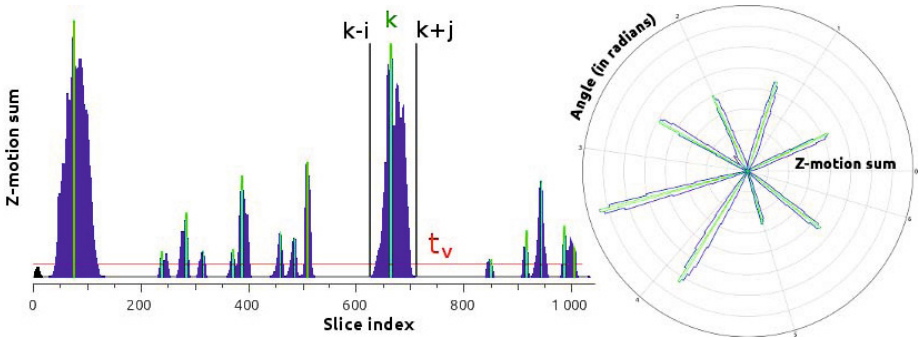
$$Z_k = |S_k - S_{k-1}|$$

The set of  $Z_k$  images provides a new 3D image of dimension  $N \times M \times K - 1$  where a big value implies a big motion. It is not a problem to have the first slice  $S_1$  without corresponding z-motion slice. It is the first or the last slice and we ignore these slices during the stem analysis: they are too noisy. The set of  $Z_k$  is used to identify slice intervals containing knots. In each of them, we identify the angular intervals containing knots.

Let us see now how to use z-motion to determine slice intervals and angular intervals containing knots.

### 2.1 Slice and Angular Intervals

The first detection identifies the slice intervals (see Fig. 2). To determine these intervals, we construct the histogram of z-motion sum (see Fig. 3). Each value represents the sum of all pixels of a slice  $Z_k$ . The peaks correspond to a slice with a lot of big motions, meaning that we are in presence of knots. The algorithm to determine significant peaks and the corresponding intervals is based on the analysis of derivative gradient. It is detailed in [9].



**Fig. 3.** Histogram and pie-chart of z-motion sum. In each of them, we identify knot intervals with the same algorithm.

Usually, the knots constitute a whorl<sup>1</sup>. This implies that there are several knots in a slice interval. To isolate each knot of a whorl, we proceed to a second analysis of z-motion in each slice interval  $[Z_{k-i}, Z_{k+j}]$ . Each slice is divided in 360 angular sectors centered on the pith. We construct the pie-chart of z-motion sum illustrated on the Fig. 3. One value corresponds to the z-motion sum on a same angular sector taken on all the slices of  $[Z_{k-i}, Z_{k+j}]$ . In the same way than for the slices, we compute intervals of angular sectors. An angular interval is a set of consecutive angular sectors containing just one knot (see Fig. 2).

### 3 A Suitable Segmentation in Angular Intervals

Let  $I$  be a billion’s image of size  $N \times M \times K$ .  $I$  can be seen as a sequence of slices  $(\mathcal{S}_k)_{k \in [1, K]}$  or as a sequence of angular sectors  $(s_d)_{d \in [1, D]}$ . After the detection process of slice intervals (see Section 2), we obtain a set  $(i_x)$  of slice intervals. The original image  $I$  restricted to an interval  $i_x$  generates a slice subsequence  $\mathcal{P}_x$  (see Fig. 4).

A detection process of angular sector is applied to each slice subsequence  $\mathcal{P}_x$ . This process furnishes a set  $(\alpha_y)$  of angular intervals with usually just one

<sup>1</sup> Knot group with the same origin, the pith, and organized in circle around the pith axis (see Fig. 1(a)).

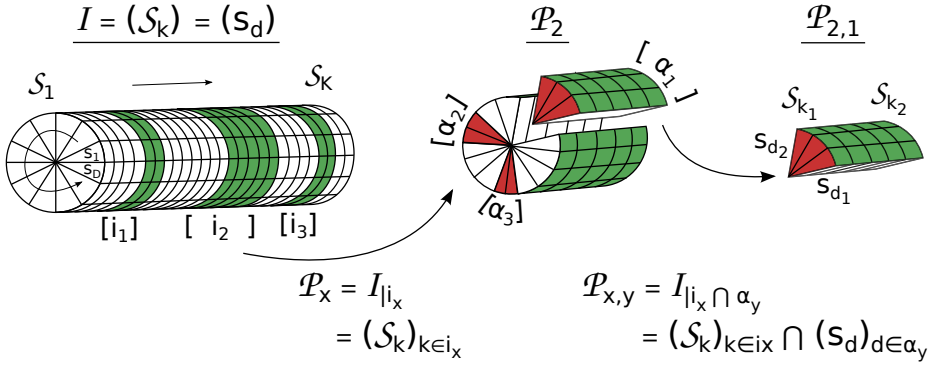


Fig. 4. Notation illustration

knot. The slice interval  $\mathcal{P}_x$  restricted to an angular interval  $\alpha_y$  is an angular subsequence  $\mathcal{P}_{x,y}$  similar to a piece of pie. In the following, we name “knot area” the subsequences  $\mathcal{P}_{x,y}$  (see on the right of the Fig. 4).

The segmentation process described in this section is applied to each knot area  $\mathcal{P}_{x,y}$ . It begins by a 3D connected component extraction based on the grey level. The objective of this extraction is to eliminate the connected components resulting from noise as growth rings. The chosen threshold is  $-100$  to be sure to not cut the knots (in general fixed between  $-90$  and  $-70$  by biologists). As a reminder, the grey level interval corresponding to the wood density in X-Ray images is  $[-900, 530]$ . As a result, we obtain one or more connected components and we keep the biggest (within the meaning of voxel number) in a new binary knot area  $\mathcal{B}_{x,y}$  with the same dimensions than  $\mathcal{P}_{x,y}$ .

The proposed algorithm is applied on each 2D slice  $S_{x,y}$  of  $\mathcal{B}_{x,y}$ . It consists of the four main steps described in the following sub-sections (see Fig. 5).

### 3.1 Step 1: 2D Connected Component Extraction

The first step consists in a 2D connected component extraction applied on  $S_{x,y}$ . It potentially contains a part of the biggest connected component of  $\mathcal{B}_{x,y}$ . We can have more than one connected component in the 2D slice  $S_{x,y}$  while  $\mathcal{B}_{x,y}$  contains only one connected component. It is a previous or a next slice of  $\mathcal{B}_{x,y}$  that merges the different connected components of  $S_{x,y}$ .

After the 2D connected component extraction, we keep the components with more than  $15^2$  pixels. We name this new slice  $S'_{x,y}$ . In Fig. 5(a), we have an example of  $S'_{x,y}$  with just one connected component drawn in pink. The two dark pink lines represents the bounds of the  $\alpha_y$  angular interval.

### 3.2 Step 2: Contour Detection

We search  $P_0$  in  $S'_{x,y}$ , the nearest pixel to the pith belonging to a 2D connected component found at the previous step. To do this, we use Andres’s circle [4]

with an increasing radius. The Andres’s circle ensures to visit all the surface of  $S'_{x,y}$  by a complete paving of plan. The found pixel  $P_0$ , drawn in red and green in Fig. 5(b), is the first point of the contour  $\mathcal{C}$ . From  $P_0$ , we applied an algorithm based on the Moore’s neighborhood to determine the other  $\mathcal{C}$  points of the nearest connected component of the pith.

To obtain better results in the next step, we smooth the contour  $\mathcal{C}$  with an averaging mask of radius 3. It is the smoothed contour  $\mathcal{C}^s$  that appears in blue in Fig. 5(b).

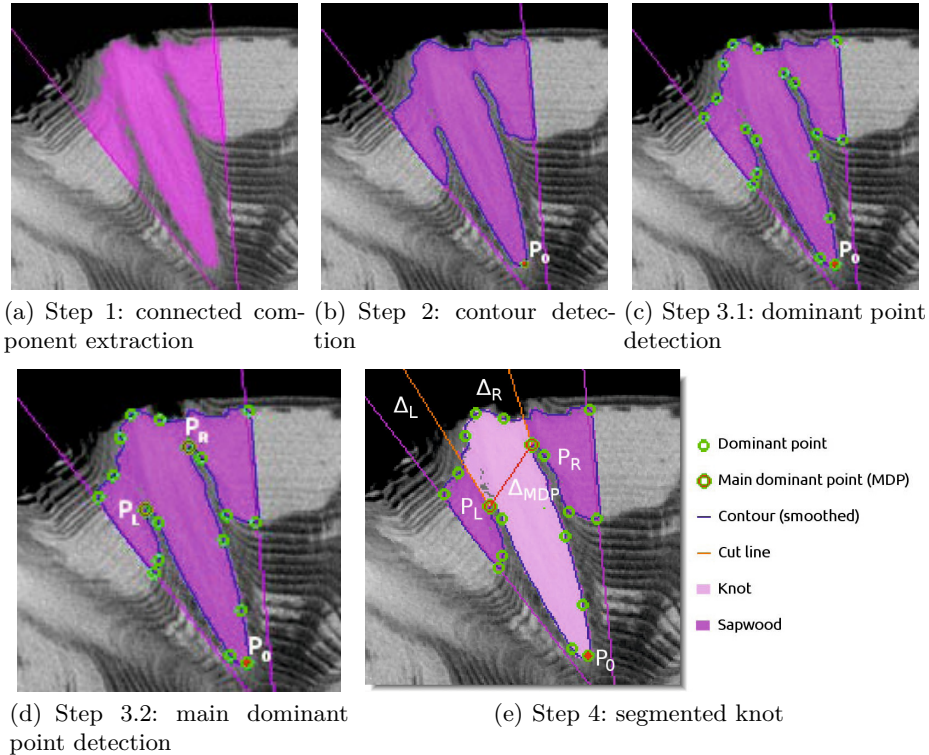


Fig. 5. Knot segmentation algorithm in four steps

### 3.3 Step 3: Dominant Points

The objective of this step is to find the junction points  $P_L$  and  $P_R$  between knot and sapwood inside the smoothed contour  $\mathcal{C}^s$ . The proposed method uses the dominant point notion (characteristic points of the contour) and a criterion based on distances to the pith to discriminate them.

**Step 3.1: Dominant Point Detection.** We detect the dominant points of  $C^s$  with a method proposed by Nguyen et al. in [13]. The algorithm relies on arithmetical discrete lines [16] and blurred segments [7].

The notion of blurred segment was introduced from the notion of arithmetical discrete line. An arithmetical discrete line, noted  $D(a, b, \mu, \omega)$ , is a set of points  $(x, y) \in \mathbb{Z}^2$  that verifies  $\mu \leq ax - by < \mu + \omega$ . A blurred segment with a main vector  $(b, a)$ , lower bound  $\mu$  and thickness  $\omega$  is a set of integer points  $(x, y)$  that is optimally bounded (see [7] for more details) by a discrete line  $D(a, b, \mu, \omega)$ . The value  $\nu = \frac{\omega-1}{\max(|a|, |b|)}$  is called the width of this blurred segment. The upper figure shows a blurred segment (the sequence of gray points) whose the optimal bounding line is  $D(5, 8, -8, 11)$ .

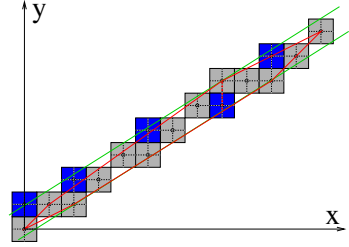


Fig. 6. A blurred segment

Nguyen et al. proposed the notion of maximal blurred segment. A maximal blurred segment of width  $\nu$  (see Fig. 6) is a blurred segment that can not be extended to the left and the right sides. A linear recognition algorithm of width  $\nu$  blurred segments [7] permits for a given contour  $C$  to obtain the sequence  $\mathcal{S}_{C,\nu}$  of all its maximal blurred segments of width  $\nu$ . We then scan the sequence  $\mathcal{S}_{C,\nu}$ : in each smallest zone of successive maximal blurred segments whose slopes are increasing or decreasing, the candidate as dominant point is detected as the middle point of this zone.

This method is used on the  $C^s$  contour detected at the previous step on the  $S'_{x,y}$  slice, with a width  $\nu = 3$ . Let  $(P_n)_{1 \leq n \leq N_P}$  be the sequence of the  $N_P$  dominant points obtained on the  $C^s$  contour (see example in Fig. 5(c)).

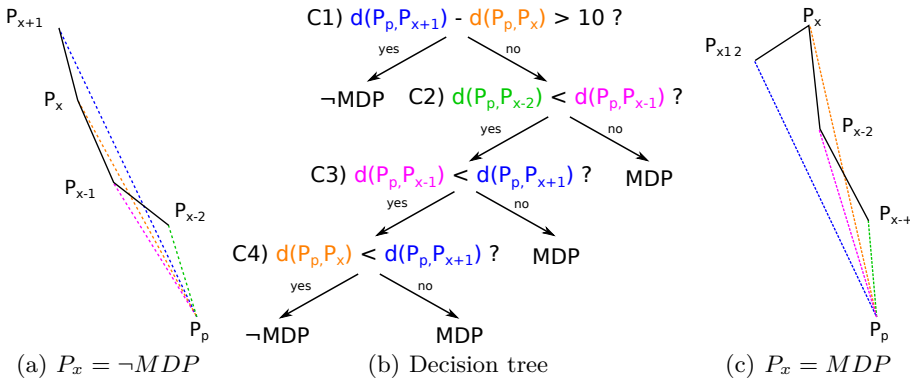


Fig. 7. Decision tree of the MDP criterion with illustration of two different executions. In (a), C1 is false and C2, C3 and C4 are true. In (c), C4 is false.

**Step 3.2: Main Dominant Points Detection.** We want to identify the two main dominant points (MDP),  $P_L$  and  $P_R$ , with  $L, R \in [1, N_P]$ ,  $L < R$  and respectively on the left and on the right of  $P_0$  (see Fig. 5(d)). The left and right sides are defined from the pith position and the orientation of the  $\alpha_y$  angular interval: left in counterclockwise, right in clockwise. We are still working on the slice  $S'_{x,y}$  and the two points make the junction between knot and sapwood on this slice.

We test successively each dominant point  $P_n$  with the decision tree presented in Fig. 7. They are tested in clockwise from  $P_1$  to determine  $P_L$  and in counterclockwise from  $P_{N_P}$  to determine  $P_R$ .

For each dominant point  $P_x$ , the MDP criterion is based on the euclidean distances  $d$  between the pith point  $P_p$  and four dominant points:

- $P_{x-2}$ ,  $P_{x-1}$ ,  $P_x$  and  $P_{x+1}$  when we search  $P_L$ ,
- $P_{x+2}$ ,  $P_{x+1}$ ,  $P_x$  and  $P_{x-1}$  when we search  $P_R$ .

The Fig. 7 presents the tree decision and examples of the MDP criterion for  $P_L$ . The first condition  $C_1$  ensures that  $P_x$  is not a MDP when the next dominant point  $P_{x+1}$  moves far enough away from the pith (more than 10 pixels) relatively to  $P_x$ . The following conditions  $C_2$  to  $C_4$  ensure a distance order such as  $d(P_p, P_{x-2}) < d(P_p, P_{x-1}) < d(P_p, P_{x+1})$ . These conditions reflect the stem shape and the pith circularity: they identify the first dominant point that does not move away from the pith.

The main dominant point detection furnishes zero, one or two points from which we can separate the knot from the pith. We need to treat the three cases to obtain a segmentation of any slice  $S'_{x,y}$ .

### 3.4 Step 4: Knot Segmentation in Sapwood from MDP

From each found MDP, we define a cut line to separate knot from sapwood. These lines are named  $\Delta_L$  for  $P_L$  and  $\Delta_R$  for  $P_R$ . Each of them is built from two points: the considered MDP and the mean point of the previous dominant points for  $P_L$ , respectively the following dominant points for  $P_R$ . Moreover, we define the segment  $\Delta_{MDP}$  linking  $P_L$  to  $P_R$  when we find two MDP (see Fig. 8(c)).

We consider three cases depending on the number of MDP, illustrated by the Fig. 8. On each of them, the light pink area corresponds to the segmented knot.

**No MDP.** In this case we considered that all the connected component of  $S'_{x,y}$  corresponds to the knot. There are two typical cases involving no MDP detection:

- when the knot and the sapwood are not connected (see Fig. 8(a)),
- when the knot is completely included in the sapwood.

**One MDP.** When there is only one MDP,  $P_L$  or  $P_R$ , the part of the connected component section corresponding to the knot is:

- on the right to  $P_L$  if  $P_L$  is the detected MDP (see Fig. 8(b)),
- on the left to  $P_R$  if  $P_R$  is the detected MDP.

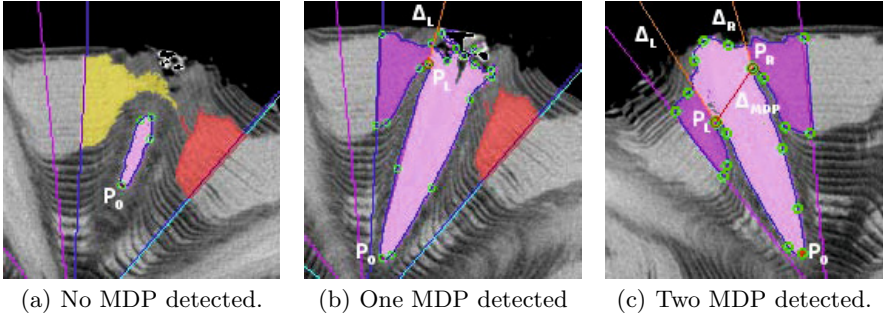


Fig. 8. Knot estimation based on the number of main dominant points

**Two MDP.** It is the most common case. It occurs most often when the knot is in contact with the sapwood. In this case, we define the segment  $\Delta_{MDP}$  which allows to separate the knot in two parts: the upper part and the lower part. They correspond respectively to the knot parts inside and outside the sapwood (see Fig. 8(c)).

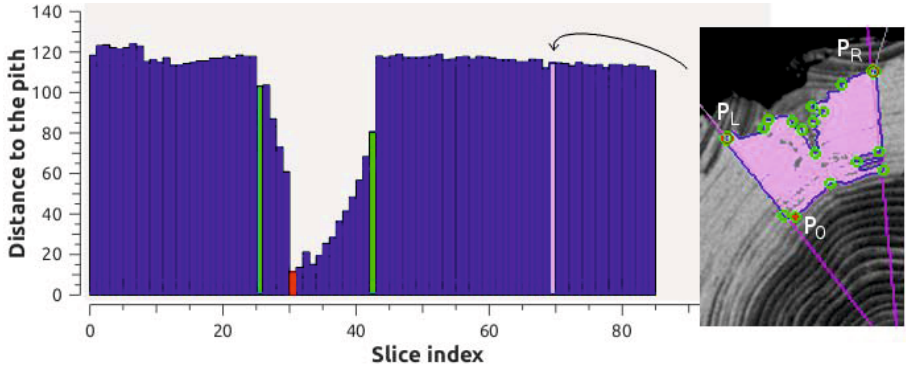
We segment separately the two parts. The lower part is segmented by defining a contour  $\mathcal{C}_1$ .  $\mathcal{C}_1$  is composed of the two parts of  $\mathcal{C}^s$ ,  $(P_R, P_0)$  and  $(P_0, P_L)$ , and the segment  $\Delta_{MDP}$ . All pixels inside  $\mathcal{C}_1$  and belonging to the considered connected component of  $S'_{x,y}$  belong to the knot. The second part, above the  $\Delta_{MDP}$  so inside the sapwood, is a restricted area of the considered connected component of  $S'_{x,y}$ . It is the connected component part simultaneously on the right of  $\Delta_L$ , on the left of  $\Delta_R$  and on the top of  $\Delta_{MDP}$ .

The fusion of the two parts produces the segmented knot of the slice  $S'_{x,y}$ , in light pink in Fig. 8(c).

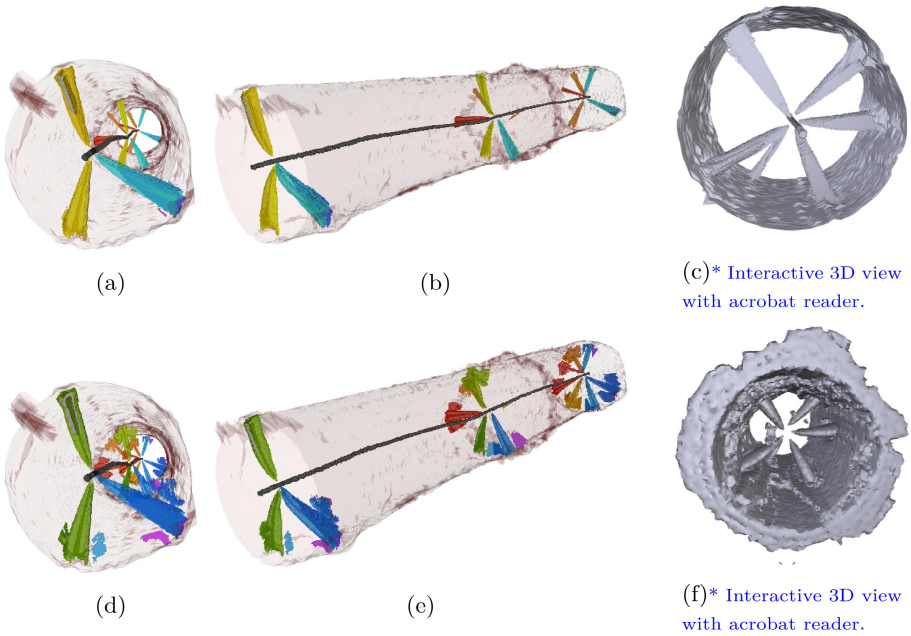
The four steps of algorithm allow to segment a knot on a 2D slice  $S_{x,y}$  of a 3D knot area  $\mathcal{P}_{x,y}$ . By merging all 2D segmented knots, we obtain the 3D reconstruction of the knot of  $\mathcal{P}_{x,y}$ . But all slices of  $\mathcal{P}_{x,y}$  do not contain part of the knot. It is necessary to detect in  $\mathcal{P}_{x,y}$  the interval  $[l, u]$  of slices containing a part of the knot to obtain a clean 3D reconstruction. In fact, the slices outside  $[l, u]$  contain just sapwood that the algorithm can segment as on the right of Fig. 9. It is usually the case but we can see that the segmented connected component does not contain knot. The  $[l, u]$  detection allows to reconstruct knot with just slices containing a part of knot.

The slice interval  $[l, u]$  is computed from the distance to the pith of the first dominant point  $P_0$  in each slice  $S_{x,y}$  of a knot area  $\mathcal{P}_{x,y}$ . We construct the corresponding histogram  $H$ , illustrated on Fig. 9. In this histogram, we can detect the knot interval  $[l, u]$ . The process starts with the localization of the minimum, in red on Fig. 9. Afterwards, we seek the bounds  $l$  and  $u$  (in green on Fig. 9) on each side of the minimum. A slice index  $j$  is the lower bound  $l$ , respectively the upper bound  $u$ , if  $H_{j-6} - H_{j-1} < 5$ , respectively if  $H_{j+6} - H_{j+1} < 5$ .



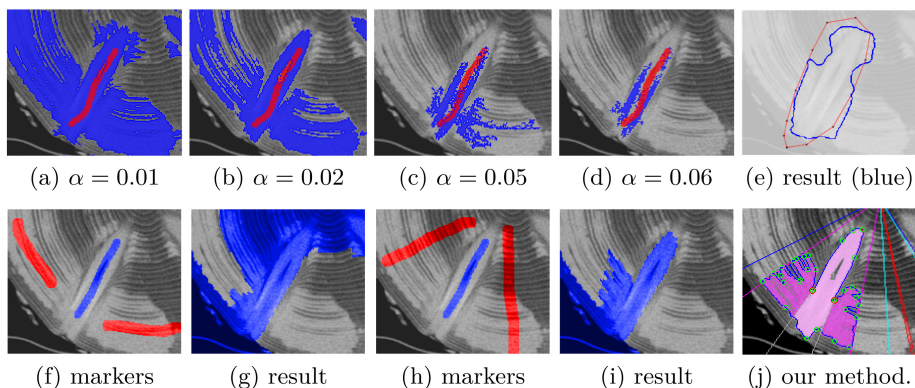


**Fig. 9.** Histogram of distances to the pith of a knot area  $\mathcal{P}_{x,y}$ . On the right, an example of sapwood segmentation in a slice without knot.



**Fig. 10.** Illustration of the segmentation results of the proposed approach (images (a,b)) and comparisons with previous work (images (d,e)) [9]. Images (c,f) show comparison between our approach (c) and deformable model (f) [10]. The static and dynamic visualizations were generated with the *DGtal* library [1].





**Fig. 11.** Experiments of various segmentation methods on the bottom left part of image of Fig. 1(b). Images (a-d) show results obtained from a Component Tree based approach [15] with different values of the user parameter  $\alpha$ . Result of morphological snake [3] is given in (e) and *power watersheds* [6] algorithm result is displayed with two configurations of markers (images (f-i)). The result of our algorithm is displayed in image (j) in light pink color.

## 4 Experiments and Comparisons

To evaluate the efficiency of our approach, the knot segmentation was applied on a difficult sample of spruce containing continue areas of sapwood. The segmentation results are presented on images (a-c) of Fig. 10. Note that we focus on the extraction of the larger branches by constraining the segmentation from a threshold on the minimal size of the segmented component. As comparison, the basic knot segmentation (images (d,e)) is performed with a simple threshold on the knot areas detected from previous work [9]. As shown in figure Fig. 10, our approach is able to remove all sapwood areas without any markers.

**Comparisons with Other Approaches.** Before defining the proposed approach, we experimented several recent and promising segmentation methods. The first one is the method of the Component Tree [15]. Even with a manual adjustment of the markers and the numeric parameter, we can see that result does not fit to the initial knot (images (a-d)) of Fig. 11. We also apply the *power watersheds* [6] and morphological snake algorithms [3] respectively on images (e) and (f-i). As comparison, the result of our segmentation method is displayed in image (j). The images (c,f) of Fig. 10 complete the comparisons of our algorithms (image (c)) with a 3D deformable model [10]. Contrary to our approach, the 3D deformable model also segments together the sapwood with the knots.

## 5 Conclusion

This paper proposes a segmentation method applied to the wood knot problem. Known as a difficult problem, we use histograms and discrete tools to propose a

new solution. The resulting approach is efficient (running time around few minutes for a complete log, without optimization) in comparison to 3D deformable model [10] (order of hours for the sample of Fig. 10). Some improvements can be done in future work to apply segmentation on large and small knots simultaneously. The source code of the algorithm is available online [8].

**Acknowledgements.** We would like to thank F. Longuetaud and F. Mothe for their wood expertise, J.-O. Lachaud and B. Taton for the source code of the deformable model [10], B. Naegel and N. Passat to provide us the component-trees based algorithm [15] and T. P. Nguyen for the dominant point detection code [13].

## References

1. DGtal: Digital geometry tools & algorithms library, <http://libdgtal.org>
2. Aguilera, C., Sanchez, R., Baradit, E.: Detection of knots using x-ray tomographies and deformable contours with simulated annealing. *Wood Res.* 53, 57–66 (2008)
3. Alvarez, L., Baumela, L., Márquez-Neila, P., Henríquez, P.: A real time morphological snakes algorithm. *Image Processing On Line* (2012)
4. Andres, E.: Discrete circles, rings and spheres. *Comp. & G.* 18(5), 695–706 (1994)
5. Andreu, J.P., Rinnhofer, A.: Modeling Knot Geometry in Norway Spruce from Industrial CT Images. In: Bigun, J., Gustavsson, T. (eds.) SCIA 2003. LNCS, vol. 2749, pp. 786–791. Springer, Heidelberg (2003)
6. Couprie, C., Grady, L., Najman, L., Talbot, H.: Power watersheds: A unifying graph-based optimization framework. *IEEE PAMI* 33(7), 1384–1399 (2010)
7. Debled-Rennesson, I., Feschet, F., Rouyer-Degli, J.: Optimal blurred segments decomposition of noisy shapes in linear time. *Comp. & Graphics* 30(1), 30–36 (2006)
8. Krähenbühl, A.: (2012), <https://github.com/adrien057/TKDetection/tags/>
9. Krähenbühl, A., Kerautret, B., Debled-Rennesson, I., Longuetaud, F., Mothe, F.: Knot Detection in X-Ray CT Images of Wood. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Fowlkes, C., Wang, S., Choi, M.-H., Mantler, S., Schulze, J., Acevedo, D., Mueller, K., Papka, M. (eds.) ISVC 2012, Part II. LNCS, vol. 7432, pp. 209–218. Springer, Heidelberg (2012)
10. Lachaud, J.O., Taton, B.: Deformable model with adaptive mesh and automated topology changes. In: *Proc. 3DIM* (2003)
11. Longuetaud, F., Leban, J.M., Mothe, F., Kerrien, E., Berger, M.O.: Automatic detection of pith on ct images of spruce logs. *Computers and Electronics in Agriculture* 44(2), 107–119 (2004)
12. Longuetaud, F., Mothe, F., Kerautret, B., Krähenbühl, A., Hory, L., Leban, J.M., Debled-Rennesson, I.: Automatic knot detection and measurements from x-ray ct images of wood: A review and validation of an improved algorithm on softwood samples. *Computers and Electronics in Agriculture* 85, 77–89 (2012)
13. Nguyen, T.P., Debled-Rennesson, I.: A discrete geometry approach for dominant point detection. *Pattern Recognition* 44(1), 32–44 (2011)
14. Nordmark, U.: Value Recovery and Production Control in the Forestry-wood Chain Using Simulation Technique. Ph.D. thesis, Luleå University of Technology (2005)
15. Passat, N., Naegel, B., Rousseau, F., Koob, M., Dietemann, J.L.: Interactive segmentation based on component-trees. *Pat. Rec.* 44(10-11), 2539–2554 (2011)
16. Reveillès, J.P.: Géométrie discrète, calculs en nombre entiers et algorithmique (Discrete geometry, integers calculus and algorithmics). Ph.D. thesis, Université Louis Pasteur, Strasbourg (1991), thèse d'état

# Multigrid Convergent Curvature Estimator

Christophe Fiorio<sup>1</sup>, Christian Mercat<sup>3</sup>, and Frédéric Rieux<sup>1,2</sup>

<sup>1</sup> LIRMM, Université Montpellier 2,  
F-34392 MONTPELLIER, France

<sup>2</sup> I3M, Université de Montpellier 2 c.c. 51  
F-34095 Montpellier Cedex 5, France

<sup>3</sup> S2HEP EA 4148 Université Claude Bernard Lyon 1  
F-69622 Villeurbanne cedex, France

**Abstract.** We propose in this paper an estimator of derivative and curvature of discrete curves. Based on adaptive convolution that preserves contour, we use local geometrical information as the heat kernel to convolve with a discrete curve and give estimation of its geometrical parameters. We recover on regular part of the curve the classical convolution based on gaussian kernel. We study the bounded error of our approach for first and second order derivative and we discuss about the multigrid convergence.

## 1 Introduction

Curvature is a geometrical invariant of shapes or curves which characterizes the object. It has a clear definition in the smooth setting but on discrete shapes, it is an important problem to give an estimation of this invariant and there exists many approaches.

A geometrical definition, based on digital segment decomposition of the shape, estimating the first derivative of the curve is given in [6,10]. Digital circle arcs decomposition is used in [3,16,15]. Decomposition into maximal segments has been proved multigrid convergent, meaning that when the digital discretization step tends to zero, the derivative estimation converges toward the underlying continuous one. This approach has been generalized by using maximal digital circle arcs in [15]. The authors studied the multigrid convergence according to the length of the digital circle arcs when the discretization step tends to zero, and conjectured the multigrid convergence of their approach. Segmentation with non-primitive objects has been studied in [14] where the authors proved the multigrid convergence. Others approaches have been proposed, based on convolution by a Gaussian kernel in [12,9] or a kernel adapted to the contour in [7]. Gaussian kernel has been widely studied in Image analysis to reduce noise effect [17]. Although Gaussian kernel is optimal on flatten parts (Theorem of Gabor [11]), its has a blurring on the contour. For this reason a large amount of works deals with adaptive kernels to reduce the blurring effect whereas still reducing the intensity of the noise. We have proposed in a previous work a digital and adaptive kernel suited for curves and surfaces [7,8], but we were not able to prove the multigrid

convergence of this approach. On the other hand, the gaussian kernel has been proved multigrid convergent for  $\mathcal{C}^3$  curves in [12] and for  $\mathcal{C}^2$  curves in [5] for derivative of order one.

We propose in this paper to study the multigrid convergence of derivatives of order one and two on curves. We rely on convolution estimators (Section 2) with Least square methods to give an estimation of the derivative (Section 3). With the help of a previous work [2], we will use the link between the two approaches to show that estimation based on adaptive kernel is multigrid convergent (Section 4), and finally we will propose numerical examples of curvature estimation on digital curves (Section 5).

## 2 Convolution Derivatives Estimator

### 2.1 Convolution with Gaussian Kernel

**Definition 1.** *Convolution Product*

Let  $F : \mathbb{Z} \rightarrow \mathbb{Z}$  and  $K : \mathbb{Z} \rightarrow \mathbb{Z}$  be two discrete functions. We call the convolution product of  $F$  by  $K$  denoted  $F * K$  the function:

$$F * K : \mathbb{Z} \rightarrow \mathbb{Z} \tag{1}$$

$$a \mapsto \sum_{i \in \mathbb{Z}} F(a - i)K(i) \tag{2}$$

**Definition 2.** *Derivative Estimation*[12]

Let  $\varphi$  be a discrete function. An estimation of the first derivative of  $\varphi$  at point  $x$  is given by:

$$(\Delta_{2m-1} * \varphi)(x) = \frac{1}{2^{2m-1}} \sum_{i=-m+1}^m \binom{2m-1}{m-1+i} (\varphi(n+i+1) - \varphi(n-1+i)) \tag{3}$$

**Theorem 3.** [5]

Let  $f : \mathbb{R} \rightarrow \mathbb{R}$  be a  $\mathcal{C}^3$  bounded function, let  $\alpha \in ]0, 1]$ ,  $K \in \mathbb{R}_+^*$  and  $h \in \mathbb{R}_+^*$ . We suppose that  $\Gamma : \mathbb{Z} \rightarrow \mathbb{Z}$  is such that  $|h\Gamma(i) - f(hi)| \leq Kh^\alpha$ . Then for  $m = h^{2(\alpha-3)/3}$  we have  $|(D_{2m-1} * u)(n) - f'(nh)| \in O(h^{2\alpha/3})$

We have proposed in [7,8] to study an adaptive kernel for derivative estimation. Gaussian Kernel previously introduced has been widely used in image analysis but its blurring effect destroy fine structures and boundaries. Nevertheless, Gabor [11] shows that on flatten part of an image gaussian kernel are the most efficient masks to reduce noise. Many works deal with an adaptive kernels that preserve contours while reducing noise [13,1]. Along these lines, we propose this discrete kernel which is gaussian on regular parts of a curve and adaptive on high curvature points.

## 2.2 Convolution with Adaptive Kernel

We recall the definition of our adaptive kernel. It is a weighted version of the classical adjacency matrix.

**Definition 4.** *Adaptive Kernel*

Let  $\Sigma \subset \mathbb{Z}^n$  be a sets of 0-connected voxels. We define on  $\Sigma$  the Markov chain in discrete time whose states are  $\mathcal{E}$ , the voxels of  $\Sigma$ , and whose transitions between two neighbors are constrained by:

- Probability  $\frac{1}{2^n}$  to go from center of the voxel to one of its corner
- Equiprobable repartitions from the corner to every adjacent voxel.

We call  $A_s$  the adjacency matrix of the adaptive digital diffusion process. Many properties of this digital diffusion process can be found in [7,8].

**Definition 5.** *First derivative estimation*

Let  $\varphi$  be a discrete function and  $A_s^m$  the adaptive kernel computed on  $\varphi$ . Its first derivative estimation at point  $n$  is given by:

$$(D_{(2m+1)}^1 * \varphi)(n) = \sum_{i=-m}^m A_s^m(n, i)(\varphi(n + i + 1) - \varphi(n - 1 + i)) \quad (4)$$

Following Theorem. 3, we will show that the estimator error is bounded and converges toward zero when the discretization step tends to zero. We use the least square method to tackle the problem.

## 3 Least Square Approximation

### 3.1 Definitions

Let  $\{y_0, y_1, \dots, y_n\}$  and  $\{x_0, x_1, \dots, x_n\}$  be two matching sets of experimental measures. We search a relation between  $x_k$  and  $y_k$  for  $k \in [1, n]$ . In the case of derivative estimation, we are looking for the line fitting the set and passing through a point  $(x_i, y_i)$   $i \in [1, n]$ . Classical least square approximation is to compute the minimum of the following sum  $S(a)$  with respect to  $a$ , the slope of the target line, with

$$S(a) = \sum_{j=1}^n ((y_j - y_i) - a(x_j - x_i))^2$$

In our situation, the diffusion process defines a reliability weight between two given points: Let  $A_s^m(i, j)$  the weight at point  $j$  starting from point  $i$ :

$$S(a) = \sum_{j=1}^n ((y_j - y_i) - a(x_j - x_i))^2 A_s^m(i, j) \quad (5)$$

Then the optimal slope  $a$  is the solution of

$$\begin{aligned} \frac{\partial S}{\partial a} &= \sum_{j=1}^n 2((y_j - y_i) - a(x_j - x_i)) \times (x_j - x_i)A_s^m(i, j) = 0 \\ &= \sum_{j=1}^n (y_j - y_i)(x_j - x_i)A_s^m(i, j) - a \sum_{j=1}^n (x_j - x_i)^2 A_s^m(i, j) = 0 \\ a &= \frac{\sum_{j=1}^n (y_j - y_i)(x_j - x_i)A_s^m(i, j)}{\sum_{j=1}^n (x_j - x_i)^2 A_s^m(i, j)} \end{aligned}$$

**Definition 6.** Let  $\mathcal{C} = \{c_0, c_1, \dots, c_n\}$  be a discrete curve and  $A_s^m$  the adaptive kernel associated with  $\mathcal{C}$ . We call the least square estimation in  $c_i = (x_i, y_i)$ , the line of slope  $a$  such that:

$$a = \frac{\sum_{j=1}^n (y_j - y_i)(x_j - x_i)A_s^m(i, j)}{\sum_{j=1}^n (x_j - x_i)^2 A_s^m(i, j)} \tag{6}$$

$$b = y_i - ax_i \tag{7}$$

In the next section, we will study the link between the least square approach and the one based on the associated weighted convolution.

### 3.2 Link with Gaussian Convolution

The classical least square derivative approach is proved to be multigrid convergent for  $\mathcal{C}^2$  curves [2]. The authors also generalize this results to second order derivative functions. We will add geometrical informations of weights in the least square approach and we will prove that the formula is related to the gaussian convolution.

**Theorem 7.** Let  $\mathcal{C} = \{c_0, c_1, \dots, c_n\}$  be a discrete curve such that  $c_j = (x_j, y_j)$  and  $A_s$  the stochastic matrix associated with  $\mathcal{C}$ . If  $A_s^m(i, j) = \frac{1}{2^{2m-1}}C_{2m+1}^{m-j+1}$ , then we have:

$$\frac{\sum_{j=0}^{2m} (y_j - y_i)(x_j - x_i)A_s^m(i, j)}{\sum_{j=0}^{2m} (x_j - x_i)^2 A_s^m(i, j)} = D_{2m-1}^1(c_i)$$

*Remark 8.* This theorem allows us to connect the gaussian weighted least square approach and the gaussian convolution. Note that the theorem only holds for gaussian weights but we will see how to deal with adaptive weights.

*Proof.* We rewrite the least square estimation formula to simplify the proof. Without loss of generality, we suppose that the estimation tangent point is centered at 0. Let  $p_j = A_s^m(i, j)$

$$a = \frac{\sum_{j=-m}^m (y_j - y_0) j p_j}{\sum_{j=-m}^m j^2 p_j}$$

If  $p_j$  is given by binomial numbers, we have:

$$a = \frac{\sum_{j=-m}^m (y_j - y_0) j C_{2m-1}^{m+j}}{\sum_{j=-m}^m j^2 C_{2m}^{m+j}}$$

We easily check that  $\sum_{j=-m}^m j^2 C_{2m}^{m+j} = m 2^{2m-1}$ .

We collect  $y_j$  in:

$$(D_{(2m+1)}^1 * \varphi)(x) = \sum_{i=0}^{2m+1} C_{2m-1}^{m+j} (y_{j+1} - y_{j-1})$$

$$\begin{aligned} C_{2m-1}^{m+j}(-y_j) + C_{2m-1}^{m+j-1}(y_j) &= y_j \left( \frac{(2m-1)!}{(m+j-1)!(m+j)!} - \frac{(2m-1)!}{(m+j)!(m-j-1)!} \right) \\ &= y_j \frac{(2m)!}{2m} \frac{m+j-(m-j)}{(m+j)!(m-j)!} \\ &= y_j \frac{1}{2m} \frac{2j(2m)!}{(m+j)!(m-j)!} \\ &= y_j \frac{j}{m} C_{2m}^{m+j} \end{aligned}$$

We will use this result to show the link with our adaptive kernel.

### 3.3 Link with Adaptive Kernel

First we need technical results about our adaptive kernel.

**Theorem 9.** [7] *Let  $\mathcal{D}(a, b, \mu, \omega)$  be a discrete line and let  $A_s$  be its adaptive kernel. Then  $\frac{1}{\sqrt{2m+1}} A_s^m(0) \xrightarrow[m \rightarrow \infty]{\mathcal{L}} \mathcal{N}(0, 1)$ .*

This theorem shows that on discrete lines, the adaptive kernel follows a normal law; the adaptive kernel has, in the limit, the same statistic distribution as the gaussian kernel.

**Proposition 10.** *The discrete standard normal diffusion, on the sampling of a curve of continuous curvature, converges in law toward the centered normalized normal law.*

More precisely:

**Lemma 11.** *Let  $f \in \mathcal{C}^2$ , Let  $x \in \mathcal{D}_f$  and  $\sigma_x$  be the standard deviation of diffusion on a discrete line of slope  $f'(x)$ . Let  $i_\epsilon$  be the discrete point associated to  $x$  on  $\varphi_\epsilon : \mathbb{Z} \rightarrow \mathbb{Z}$  discretization of  $f$  with a step  $\epsilon$ ,  $A_s$  the convolution kernel associated to  $\varphi_\epsilon$  and  $p_\epsilon$  the length of the maximal segment centered at  $i_\epsilon$ . We choose  $m_\epsilon$  such that  $\frac{p_\epsilon}{2} > m_\epsilon > \frac{p_\epsilon}{4}$  and we note  $X_\epsilon(y) = \frac{1}{m_\epsilon \sigma_x} A_s^{m_\epsilon}(i_\epsilon, i_\epsilon + \lfloor \frac{ym_\epsilon \sigma_x}{\epsilon} \rfloor)$ , the standard normal diffusion process at  $i_\epsilon$ . Then  $(X_\epsilon)_{\epsilon \in \mathbb{R}}$  converges in law toward the standard normal law:*

$$X_\epsilon \xrightarrow[\epsilon \rightarrow 0]{\mathcal{L}} \mathcal{N}(0, 1)$$

*Remark 12.* This proposition refers to the asymptotic distribution of the diffusion process for large time being equivalent to a normal law for a given standard deviation. Therefore, on a curve of continuous curvature, the adaptive kernel converges toward the gaussian kernel.

*Proof.* The starting point of diffusion  $i_\epsilon$  belong to  $S$ , maximal segment centered at  $i_\epsilon$  with length  $p_\epsilon$ . Statistical distribution of weights starting from point  $i_\epsilon$  have a standard deviation  $\sigma_\epsilon$ . Because of the class of the function  $f \in \mathcal{C}^2$ , the deviation  $\sigma_\epsilon$  actually converges, when  $\epsilon \rightarrow 0$ , toward a number  $\sigma_x$ , which is the standard deviation of the discrete diffusion on the discrete line of slope  $f'(x)$ .

For  $j$  such that  $|i_\epsilon - j| < p_\epsilon$ , being on a maximal segment, the standard deviation of the process between  $i_\epsilon$  and  $j$  for a time  $m < p_\epsilon$  is identical to the process on the discrete line  $S$ .

According to theorem. 9, the standard normal diffusion process on the discrete line  $S$  converges in law toward the normal law when  $m \rightarrow \infty$ . And according to [10],  $p_\epsilon$  is not bounded when  $\epsilon \rightarrow 0$ . Then for  $\frac{p_\epsilon}{2} > m_\epsilon > \frac{p_\epsilon}{4}$ ,

$$X_\epsilon \xrightarrow[\epsilon \rightarrow 0]{\mathcal{L}} \mathcal{N}(0, 1)$$

## 4 Multigrid Convergence of Derivatives Estimator

In this section, we study the asymptotic convergence of the derivative estimator of order one and two. In the last section, we study the link between convolution and least square approach. We will use this link to prove the multigrid convergence of our approach. In [2], the authors propose a proof of multigrid convergence of least square approach without ponderation. We extend their work to the gaussian approach and our adaptive approach. The need for this extension has been already documented in image analysis: the constant averaging mask is not efficient, the gaussian mask taking into account the distance from the treated pixel is better, and to preserve contours, one needs an adaptive mask that takes into account the geometry of the image. We propose a similar approach on discrete curves, and in a forthcoming paper we will apply it to gray level images.



### 4.1 Theoretical Convergence

**Theorem 13.** Let  $y_i = D^1(f)(x_i), x_i \in \{x_1, x_2, \dots, x_n\}$  be the discretisation of a function  $y = f(x) \in \mathcal{C}^2$ . Then  $\forall j, \exists \zeta_j \in [x, x_j]$  such that:

$$\forall k, |D^1(f)(x_k) - f'(x_k)| \leq \max_{x_j \in \mathcal{V}_k} \left\{ \left| \frac{f''(\zeta_j)}{2}(x_j - x_k) \right| \right\}$$

*Proof.* Under the hypothesis that  $f$  is  $\mathcal{C}^2$ , we have a Taylor expansion in  $x_k$ :  $\forall x \in \mathbb{R}, \exists \zeta \in [x, x_k]$

$$f(x) = f(x_k) + f'(x_k)(x - x_k) + \frac{f''(\zeta)}{2}(x - x_k)^2$$

$\forall j, \exists \zeta_j$  with a weight  $p_j$  such that:

$$y_j - y_k = f(x_j) - f(x_k) = f'(x_k)(x_j - x_k) + \frac{f''(\zeta_j)}{2}(x_j - x_k)^2 \tag{8}$$

Bringing in Equation 8 into the definition of the derivative, we get:

$$\begin{aligned} D^1(f)(x_k) &= \frac{\sum_{j=1}^n (y_j - y_k)(x_j - x_k)p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j} \\ &= \frac{\sum_{j=1}^n f'(x_k)(x_j - x_k)^2 p_j + \frac{f''(\zeta_j)}{2}(x_j - x_k)^3 p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j} \\ &= f'(x_k) + \frac{\sum_{j=1}^n \frac{f''(\zeta_j)}{2}(x_j - x_k)^3 p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j}. \end{aligned}$$

Then,

$$\begin{aligned} |D^1(f)(x_k) - f'(x_k)| &= \left| \frac{\sum_{j=1}^n \frac{f''(\zeta_j)}{2}(x_j - x_k)^3 p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j} \right| \\ &\leq \frac{\sum_{j=1}^n \max_{j \in \mathcal{V}_k} \left\{ \left| \frac{f''(\zeta_j)}{2}(x_j - x_k) \right| \right\} (x_j - x_k)^2 p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j} \\ &\leq \max_{j \in \mathcal{V}_k} \left\{ \left| \frac{f''(\zeta_j)}{2}(x_j - x_k) \right| \right\}. \end{aligned}$$

With a similar approach, we have a bounded error for derivatives of order two:

**Theorem 14.** Let  $y_i = D^2(f)(x_i), x_i \in \{x_1, x_2, \dots, x_n\}$  be the discretization of a function  $y = f(x)$  defined on  $I \in \mathcal{C}^3$ . We note  $h$  the discretization step and  $\Delta_{ik} = \frac{(x_i - x_k)^3 p_i}{\sum_{i=1}^n (x_i - x_k)^2 p_i}$ . Then  $\forall i, j, k, \exists \zeta_j \in [x, x_j], \exists \epsilon_i \in [x, x_i]$  et  $\exists \epsilon'_i \in [x, x'_i]$  such that:

$$|D^2(f)(x_k) - f''(x_k)| \leq \max_{j \in \mathcal{V}_k} \left\{ \left| \frac{f'''(\zeta_j)}{2}(x_j - x_k) \right| \right\} + \max_{i, j \in \mathcal{V}_k} \left\{ \frac{n}{(x_j - x_k)h} f''(\epsilon_i) \Delta_{ik} - f''(\epsilon'_i) \Delta_{ij} \right\}$$

*Proof.* Let  $x_k \in I$ , the second derivative at point  $k$  is given by:

$$D^2(f)(x_k) = \frac{\sum_{j=1}^n (D^1(f)(x_j) - D^1(f)(x_k))(x_j - x_k)p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j} \tag{9}$$

Identically to the last proof,  $\exists \zeta_j \in [x, x_j]$  such that:

$$f'(x_j) - f'(x_k) = f''(x_k)(x_j - x_k) + \frac{f'''(\zeta_j)}{2!}(x_k - x_j)^2 \tag{10}$$

Identically to Theorem 13, we have:

$$\begin{aligned} \exists \zeta_i \in [x, x_i], \quad D^1(f)(x_k) &= f'(x_k) + \frac{\sum_{i=1}^n \frac{f''(\zeta_i)}{2}(x_i - x_k)^3 p_i}{\sum_{i=1}^n (x_i - x_k)^2 p_i} \\ \exists \zeta'_i \in [x, x'_i], \quad D^1(f)(x_j) &= f'(x_j) + \frac{\sum_{i=1}^n \frac{f''(\zeta'_i)}{2}(x_i - x_j)^3 p_i}{\sum_{i=1}^n (x_i - x_j)^2 p_i} \end{aligned} \tag{11}$$

$$\begin{aligned} f'(x_j) - f'(x_k) &= f''(x_k)(x_j - x_k) + \frac{f'''(\zeta_j)}{2}(x_j - x_k)^2 \\ (f'(x_j) - f'(x_k))(x_j - x_k) &= f''(x_k)(x_j - x_k)^2 + \frac{f'''(\zeta_j)}{2}(x_j - x_k)^3 \end{aligned}$$

$$\begin{aligned} (D^1(f)(x_j) - D^1(f)(x_k))(x_j - x_k) &= f''(x_k)(x_j - x_k)^2 + \frac{f'''(\zeta_j)}{2}(x_j - x_k)^3 + (x_j - x_k) \\ &\quad \left\{ \frac{\sum_{i=1}^n \frac{f''(\zeta_i)}{2}(x_i - x_k)^3 p_i}{\sum_{i=1}^n (x_i - x_k)^2 p_i} - \frac{\sum_{i=1}^n \frac{f''(\zeta'_i)}{2}(x_i - x_j)^3 p_i}{\sum_{r=1}^n (x_i - x_j)^2 p_i} \right\} (x_j - x_k) \end{aligned} \tag{12}$$

Bringing Equation 12 into Equation 9,

$$D^2(f)(x_k) = \frac{\sum_{j=1}^n (D^1(f)(x_j) - D^1(f)(x_k))(x_j - x_k)p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j} \tag{13}$$

$$\tag{14}$$

$$\begin{aligned} D^2(f)(x_k) &= \frac{\sum_{j=1}^n (f''(x_k)(x_j - x_k)^2 p_j + \frac{f'''(\zeta_j)}{2}(x_j - x_k)^3 p_j)}{\sum_{j=1}^n (x_j - x_k)^2 p_j} + \\ &\quad \frac{\sum_{j=1}^n \left\{ \frac{\sum_{i=1}^n \frac{f''(\zeta_i)}{2}(x_i - x_k)^3 p_i}{\sum_{i=1}^n (x_i - x_k)^2 p_i} - \frac{\sum_{i=1}^n \frac{f''(\zeta'_i)}{2}(x_i - x_j)^3 p_i}{\sum_{i=1}^n (x_i - x_j)^2 p_i} \right\} (x_j - x_k)p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j} \end{aligned}$$

$$\begin{aligned}
 D^2(f)(x_k) - f''(x_k) &= \frac{\sum_{j=1}^n \frac{f'''(\zeta_j)}{2} (x_j - x_k)^3 p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j} \\
 &\quad + \frac{\sum_{j=1}^n \left\{ \frac{\sum_{i=1}^n \frac{f''(\zeta_j)}{2} (x_i - x_k)^3 p_i}{\sum_{i=1}^n (x_i - x_k)^2 p_i} - \frac{\sum_{i=1}^n \frac{f''(\zeta'_j)}{2} (x_i - x_j)^3 p_i}{\sum_{i=1}^n (x_i - x_j)^2 p_i} \right\} (x_j - x_k) p_j}{\underbrace{\sum_{j=1}^n (x_j - x_k)^2 p_j}_{S_2}} \\
 S_2 &= \frac{\sum_{j=1}^n \left\{ \frac{\sum_{i=1}^n \frac{f''(\zeta_j)}{2} (x_i - x_k)^3 p_i}{\sum_{i=1}^n (x_i - x_k)^2 p_i} - \frac{\sum_{i=1}^n \frac{f''(\zeta'_j)}{2} (x_i - x_j)^3 p_i}{\sum_{i=1}^n (x_i - x_j)^2 p_i} \right\} (x_j - x_k) p_j}{\sum_{j=1}^n (x_j - x_k)^2 p_j}
 \end{aligned}$$

We note  $h$  the discretization step. We call  $(x_i - x_k) = \delta_{ik}h$ :

$$S_2 = \frac{\sum_{j=1}^n \delta_{jk} h p_j \left\{ \frac{\sum_{i=1}^n \frac{f''(\zeta_j)}{2} \delta_{ik}^3 h^3 p_i}{\sum_{i=1}^n \delta_{ik}^2 h^2 p_i} - \frac{\sum_{i=1}^n \frac{f''(\zeta'_j)}{2} \delta_{ij}^3 h^3 p_i}{\sum_{i=1}^n \delta_{ij}^2 h^2 p_i} \right\}}{\sum_{j=1}^n \delta_{jk}^2 h^2 p_j}$$

We lay  $\Delta_{ik} = \frac{\delta_{ik}^3 p_i}{\sum_{i=1}^n \delta_{ik}^2 p_i}$

$$S_2 = \frac{\sum_{j=1}^n \delta_{jk} h p_j \{ \sum_{i=1}^n f''(\zeta_i) \Delta_{ik} - f''(\zeta'_i) \Delta_{ij} \}}{\sum_{j=1}^n \delta_{jk}^2 h^2 p_j}$$

By taking the maximum:

$$\begin{aligned}
 S_2 &\leq \max_{i,j \in I} \left\{ \frac{\sum_{j=1}^n \frac{\delta_{jk}^2 h^2 p_j}{\delta_{jk} h} \{ \sum_{i=1}^n f''(\zeta_i) \Delta_{ik} - f''(\zeta'_i) \Delta_{ij} \}}{\sum_{j=1}^n \delta_{jk}^2 h^2 p_j} \right\} \\
 &\leq \max_{i,j \in \mathcal{V}_k} \left\{ \left| \frac{n}{\delta_{ik} h} (f''(\zeta_i) \Delta_{ik} - f''(\zeta'_i) \Delta_{ij}) \right| \right\}
 \end{aligned}$$

We deduce the bounded error for second order derivative:

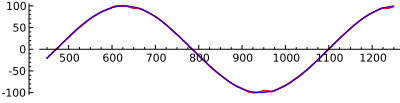
$$|D^2(f)(x_k) - f''(x_k)| \leq \max_{j \in \mathcal{V}_k} \left\{ \left| \frac{f'''(\zeta_j)}{2} (x_j - x_k) \right| \right\} + \max_{i,j \in \mathcal{V}_k} \left\{ \left| \frac{n}{\delta_{ik} h} f''(\zeta_i) \Delta_{ik} - f''(\zeta'_i) \Delta_{ij} \right| \right\}$$

**Corollary 15.** *Let  $y_i = D^1(f)(x_i), x_i \in \{x_1, x_2, \dots, x_n\}$  be the discretization of a function  $y = f(x)$  with  $\in C^3$ . Let  $h$  be the discretization step. Then, when  $h \rightarrow 0$ , we have  $|D^1(f)(x_k) - f'(x_k)| \rightarrow 0$  and  $|D^2(f)(x_k) - f''(x_k)| \rightarrow 0$*

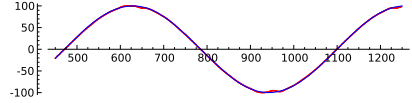
*Proof.* The study of the convergence of the bounded error could be found in [2]

### 4.2 Experimentation

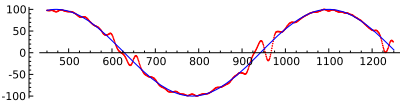
In this section, we propose examples to support the convergence of this estimator. We propose an estimation of the first derivative in Figure 1. We plot as well the estimation of the second derivative with a mask of length 50 and a discretization step of  $\frac{1}{100}$  (Figure 3). Then we study the convergence of this estimation when the mask length is increased and the discretization step decreased. Whereas, in the first approximation there are some artifacts, we can see for a larger mask and a better digitalization step that we recover a good approximation (Figure 5 and Figure 6).



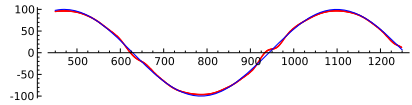
**Fig. 1.** First order derivative estimation for a mask of length 50 for  $x \mapsto \sin(x)$



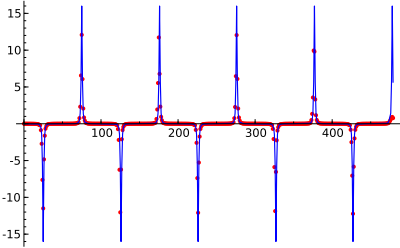
**Fig. 2.** First order derivative estimation for a mask of length 150 for  $x \mapsto \sin(x)$



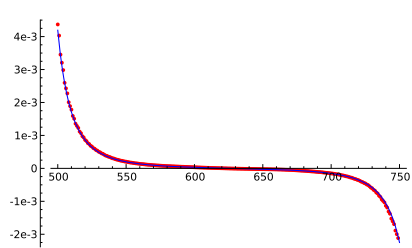
**Fig. 3.** Second order derivative estimation for a mask of length 50 for  $x \mapsto \sin(x)$



**Fig. 4.** Second order derivative estimation for a mask of length 600 for  $x \mapsto \sin(x)$



**Fig. 5.** Curvature estimation with a mask of length 300 and discrete step  $h = \frac{1}{100}$



**Fig. 6.** Zoom on curvature estimation details Figure. 5

**Fig. 7.** Examples of curvature estimation on the function  $x \mapsto \sin(x)$ . Even for quite large discretization steps and small mask size, we have a reasonably good approximation of the curvature (Figure 5). But when considering smaller discretization steps, the estimation converges toward the exact values.

## 5 Conclusion

In this article, we proposed a curvature estimator based on an adaptive kernel. Our approach is similar to a gaussian convolution on regular parts of the curve. First we have studied the link between the gaussian kernel and our approach. We proved that the gaussian process converges toward our adaptive estimator when the grid step converges toward zero and the mask length converges toward infinity. Then we used a least square method to bound the error between the estimation and the exact function with adaptive approach. This bounded error converges toward zero when the grid step converges toward zero and the length mask converges toward infinity. The issue of the length of the mask has not been studied in this paper and we have few information about the minimal length to have a satisfying estimation. In a forthcoming paper we will study the minimal length mask for a target bound of the error.

## References

1. Alvarez, L., Lions, P.L., Morel, J.M.: Image selective smoothing and edge detection by nonlinear diffusion. ii. *SIAM Journal on Numerical Analysis* 29(3), 845–866 (1992)
2. An, Y., Shao, C., Wang, X., Li, Z.: Geometric properties estimation from discrete curves using discrete derivatives. *Computers & Graphics* 35(4), 916–930 (2011)
3. Coeurjolly, D., Miguet, S., Tougne, L.: Discrete Curvature Based on Osculating Circle Estimation. In: Arcelli, C., Cordella, L.P., Sanniti di Baja, G. (eds.) *IWVF 2001*. LNCS, vol. 2059, pp. 303–312. Springer, Heidelberg (2001)
4. Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.): *DGCI 2008*. LNCS, vol. 4992. Springer, Heidelberg (2008)
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., Tougne, L.: Optimal Time Computation of the Tangent of a Discrete Curve: Application to the Curvature. In: Bertrand, G., Couprie, M., Perroton, L. (eds.) *DGCI 1999*. LNCS, vol. 1568, pp. 31–40. Springer, Heidelberg (1999)
7. Fiorio, C., Mercat, C., Rieux, F.: Curvature Estimation for Discrete Curves Based on Auto-adaptive Masks of Convolution. In: Barneva, R.P., Brimkov, V.E., Hauptman, H.A., Natal Jorge, R.M., Tavares, J.M.R.S. (eds.) *CompIMAGE 2010*. LNCS, vol. 6026, pp. 47–59. Springer, Heidelberg (2010)
8. Fiorio, C., Mercat, C., Rieux, F.: Adaptive Discrete Laplace Operator. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Wang, S., Kyungnam, K., Benes, B., Moreland, K., Borst, C., DiVerdi, S., Yi-Jen, C., Ming, J. (eds.) *ISVC 2011, Part II*. LNCS, vol. 6939, pp. 377–386. Springer, Heidelberg (2011)
9. Fourey, S., Malgouyres, R.: Normals and curvature estimation for digital surfaces based on convolutions. In: Coeurjolly, et al. (eds.) [4], pp. 287–298
10. Lachaud, J.-O., Vialard, A., de Vieilleville, F.: Fast, accurate and convergent tangent estimation on digital contours. *Image Vision Comput.* 25(10), 1572–1587 (2007)
11. Lindenbaum, M., Fischer, M., Bruckstein, A.M.: On gabor's contribution to image enhancement. *Pattern Recognition* 27(1), 1–8 (1994)

12. Malgouyres, R., Brunet, F., Fourey, S.: Binomial convolutions and derivatives estimation from noisy discretizations. In: Coeurjolly, et al. (eds.) [4], pp. 370–379
13. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.* 12(7), 629–639 (1990)
14. Provot, L., Gérard, Y.: Estimation of the Derivatives of a Digital Function with a Convergent Bounded Error. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) DGCI 2011. LNCS, vol. 6607, pp. 284–295. Springer, Heidelberg (2011)
15. Roussillon, T., Lachaud, J.-O.: Accurate Curvature Estimation along Digital Contours with Maximal Digital Circular Arcs. In: Aggarwal, J.K., Barneva, R.P., Brimkov, V.E., Koroutchev, K.N., Korutcheva, E.R. (eds.) IWCI 2011. LNCS, vol. 6636, pp. 43–55. Springer, Heidelberg (2011)
16. Roussillon, T., Sivignon, I., Tougne, L.: Measure of circularity for parts of digital boundaries and its fast computation. *Pattern Recognition* 43(1), 37–46 (2010)
17. Weickert, J.: Anisotropic diffusion in image processing. Kaiserslautern (1996)

# Author Index

- Abdmouleh, Fatma 277  
Andres, Eric 241, 253  
Arcelli, C. 131  
Ayala, Dolors 143
- Batenburg, K. Joost 372  
Berthé, Valérie 107  
Bertrand, Gilles 71, 83  
Bezerra, Nivando 71  
Biri, Venceslas 119  
Bodini, O. 95  
Brunetti, Sara 288
- Cerri, Andrea 180, 192  
Chassery, Jean-Marc 1  
Chaussard, John 119  
Ciria, J.C. 59  
Coeurjolly, David 215  
Čomić, Lidija 323  
Couprie, Michel 71, 119, 360  
Cruz-Matías, Irving 143
- Debled-Rennesson, I. 383  
De Florian, Leila 323  
Domínguez, E. 59  
Duchon, Ph. 95  
Dulio, Paolo 288  
Dyshkant, Natalia 47
- Esbelin, Henri-Alex 335  
Ethier, Marc 192  
Evenou, Pierre 169
- Fiorio, Christophe 395  
Fortes, Wagner 372  
Francés, A.R. 59  
Frosini, Andrea 300, 311  
Frosini, Patrizio 192  
Fu, Xinbo 228
- Gonzalez, Damien 335
- Hoarau, André 35
- Iuricich, Federico 323
- Jacquot, A. 95  
Jamet, Damien 107  
Janaszewski, Marcin 360  
Jolivet, Timo 107
- Kenmochi, Yukiko 155, 253  
Kerautret, Bertrand 265, 383  
Klette, Gisela 228  
Klette, Reinhard 228  
Krähenbühl, A. 383  
Kudelski, Dimitri 203
- Lachaud, Jacques-Olivier 215  
Landi, Claudia 180  
Largeteau-Skapin, Gaëlle 241  
Levallois, Jérémy 215  
Li, Fajie 228
- Malgouyres, Rémy 335  
Mari, Jean-Luc 203  
Mercat, Christian 395  
Monteil, Thierry 35  
Mutafchiev, L. 95
- Ngo, Phuc 155  
Noël, Laurent 119  
Normand, Nicolas 169
- Passat, Nicolas 155  
Peri, Carla 288  
Phan, Minh Son 253  
Picouveau, Christophe 300, 311  
Postolski, Michał 360  
Provençal, Xavier 107
- Quintero, A. 59
- Rebatel, Fabien 11  
Rieux, Frédéric 395  
Rinaldi, Simone 300
- Samir, Chafik 335  
Sanniti di Baja, G. 131  
Serino, L. 131  
Sivignon, Isabelle 1, 23  
Slembrouck, Maarten 347

Strand, Robin 169  
Sugimoto, Akihiro 253  
Tajine, Mohamed 277  
Talbot, Hugues 155, 253  
Thiel, Édouard 11  
Toutant, Jean-Luc 265

Vacavant, Antoine 265  
Van Haerenborgh, Dirk 347  
Veelaert, Peter 347  
Viseur, Sophie 203  
Zrour, Rita 241, 253